Rate Control for Predictive Transform Screen Content Video Coding based on RANSAC

Victor Sanchez, *Member, IEEE*

***Abstract*—In predictive transform video coding, optimal bit allocation and quantization parameter (QP) estimation are important to control the bit rate of blocks, frames and the whole sequence. Common solutions to this problem rely on trained models to approximate the rate-distortion (R-D) characteristics of the video content during coding. Moreover, these solutions are mainly targeted for natural content sequences, whose characteristics differ greatly from those of screen content (SC) sequences. In this paper, we depart from such trained R-D models and propose a low-complexity RC method for SC sequences that leverages the availability of information about the R-D characteristics of previously coded blocks within a frame. Namely, our method first allocates bits at the frame- and block-levels based on their motion and texture characteristics. It then approximates the R-D and R-QP curves of each block by a set control points and random sample consensus (RANSAC). Finally, it computes the appropriate block-level QP values to attain a target bit rate with the minimum distortion possible. The proposed RC method is embedded into a standard High-Efficiency Video Coding (H.265/HEVC) encoder and evaluated on several SC sequences. Our results show that our method not only attains better R-D performance than that of H.265/HEVC and other methods designed for SC sequences but also attains a more constant and higher reconstruction quality on all frames.**

***Index Terms*—video coding, rate control, RANSAC, block-level bit allocation, HEVC.**

I. Introduction

Rate control (RC) methods are important to efficiently constraint the bit rate during video coding to satisfy any channel bandwidth, end-to-end delay or storage requirements [1–3]. In essence, RC aims at minimizing the distortion of the coded video data subject to a bit rate constraint. This is usually achieved by 1) employing optimal bit allocation (OBA) schemes that minimize the distortion of the reconstructed video, and 2) determining the appropriate quantization parameter (QP) that ensures that the target bit rate is attained. For the case of predictive transform coding (PTC), OBA can be done at different coding levels: Group-Of-Picture (GOP)-level [4], frame-level [5] and block-level [6, 7]. Usually, the finer the level, the higher the RC accuracy. In this work, we not only focus on frame-level OBA, but also on block-level OBA and QP-estimation because it has been shown that coding blocks at the appropriate bit rate with the minimum distortion possible helps to further improve the coding performance on subsequent blocks, frames, and the whole video sequence [8].

This work is supported by the EU Horizon 2020 - Marie Sklodowska-Curie Actions through the project Computer Vision Enabled Multimedia Forensics and People Identification (Project No. 690907, Acronym: IDENTITY). V. Sanchez is with the Dept. of Computer Science, University of Warwick, U.K. This work was done while he was a Visiting Researcher at the School of Electrical & Information Engineering, University of Sydney, Australia (e-mail: v.f.-sanchez-silva@warwick.ac.uk)

Block-level OBA and QP estimation are particularly challenging because of the frame-level dependencies induced by inter-prediction and the block-level dependencies induced by intra-prediction. However, if one assumes that the blocks are coded independently, it is possible to distribute a bit budget independently among the constituent blocks of each frame. Such an assumption also allows estimating a QP value for each block according to a bit budget. Lagrangian optimization can then be used to achieve the minimum average distortion under a given bit constraint. In practice, the dependencies among frames and blocks may reduce the efficiency of Lagrangian optimization because the rate-distortion (R-D) characteristics of a block are unknown before its reference blocks are coded. Dynamic programming can be employed in these cases at the expense of considerably increasing the complexity of the optimization problem [9]. A common strategy to reduce such complexity is to use R-D models to approximate OBA solutions before the actual coding. Such models include those based on ρ-domains [10, 11], polynomials [12–15], logarithms [16], the slope of a reference R-D curve [1], and more recently those learned through machine learning techniques [17, 18].

The low computational complexity of RC methods based on R-D models has fuelled their widespread use in modern video codecs. For example, the reference implementation of the H.265/HEVC standard includes RC as a non-normative part. This RC approach is based on a rate (R)-λ model to approximate the slope of the R-D curve of the video sequence to be coded. Such an R-D model, which is usually computed based on training data, is effective for video sequences that share the same characteristics as those of the training data, e.g., similar textures and motion. However, it may fail for sequences that significantly differ from the training data. For example, the R-D model obtained from training sequences depicting natural content (NC) may perform poorly on screen content (SC) sequences. One simple way to address this problem is by defining a distinct R-D model for each type of content. This is not practical, as new imaging technologies are likely to introduce new content with a wide range of characteristics, thus requiring pre-computing a large number of R-D models. An alternative solution is to design a recursive R-D model for the sequence to be coded, such as those proposed for frame-level OBA in [4]. Although such solutions do not require computing R-D models from a set of training sequences, they still involve recomputing a model, thus increasing the overall complexity.

This paper proposes a low-complexity RC method that leverages the availability of information about the R-D characteristics of previously coded blocks within a frame based on the promising results obtained in [3]. Our RC method

focuses on frame-level OBA, block-level OBA and block-level QP estimation within the context of PTC of SC sequences. Specifically, it allocates bits to each frame based on its texture and relative motion with respect to previously coded frames. At the block-level, bit allocation is performed based on the coding complexity of the blocks. The method then linearly approximates the R-D and Rate-QP (R-QP) curves of each block based on a set of control points and computes the appropriate QP value to attain a target bit rate with the minimum distortion possible. Compared to our previous work in [3], our RC method presents four main contributions. First, it incorporates a frame-level bit allocation strategy specifically tailored for SC sequences. Second, it increases the approximation accuracy of the block-level R-D curves by using random sample consensus (RANSAC), which allows eliminating any R-D information considered to be an outlier. Third, it approximates the block-level R-QP curves by using a set of control points and RANSAC to accurately estimate QP values. And fourth, it is designed not only for intra-coded frames but also for inter-coded frames. To the best of our knowledge, no other RC method has been proposed based on the linear approximation of both, the R-D and R-QP curves of video sequences at the block-level. We integrate our RC method into a standard H.265/HEVC encoder for the coding of several SC sequences using both intra-coded and inter-coded frames. Performance evaluation results show that the proposed RC method attains the overall target bit rate very accurately with important improvements on R-D performance.

The rest of the paper is organized as follows. Section II briefly reviews the related work on RC for SC sequences. We describe our RC method in Section III. Section IV presents our performance evaluations and discusses the results. Finally, Section V concludes this paper.

II. Rate control for SC sequences

Within the context of PTC of video sequences, RC aims at attaining, as accurately as possible, a target bit rate, R_{target}, with a minimum distortion, D [19]:

$$\min D \quad \text{subject to} \quad R \leq R_{target}. \tag{1}$$

Such an optimization can be handled as an unconstrained rate-distortion optimization (RDO) by minimizing the total R-D cost function, J [19, 20]:

$$J(T_0, T_1, \ldots, T_{N-1}) = \sum_{i=0}^{N-1} D_i + \lambda * \sum_{i=0}^{N-1} R_i, \tag{2}$$

where T_i is the optimal number of bits assigned to entity i within a coding level, which minimizes distortion D_i between the original and the reconstructed i^{th} entity subject to a bit budget, R_i. The Lagrangian multiplier, λ, controls the trade-off between rate and distortion by adjusting the number of bits spent on each entity. Hence, the attained bit rate depends not only on the selected QP, but also on λ [21].

It has been shown in [22] that the R-D characteristics of NC video sequences can be approximated as a hyperbolic function of the form:

$$D(R) = C * R^{-K}, \tag{3}$$

where C and K are determined by parameter fitting on an experimentally obtained R-D curve of the sequence. The Lagrangian multiplier, λ, in Eq. 2, which corresponds to the slope of the R-D curve, can then be obtained by differentiating Eq. 3 w.r.t. R [21, 22]:

$$\lambda = -\frac{\partial D}{\partial R} = C * K * R^{(-K-1)} = \alpha R_{bpp}^{\beta}, \tag{4}$$

where $\alpha = C * K$, $\beta = -K - 1$, and R_{bpp} is the target bit rate in *bits per pixel* (bpp). Eq. 4 is the R-D model used by the reference implementation of the H.265/HEVC standard for RC, also known as the R-λ model. Since computing the R-D curve of each sequence to be coded is not practical, α and β are determined based on training data. Once λ is computed, QP values at the largest coding unit (LCU) and frame levels are obtained according to the linear model [23]:

$$QP = a\log\lambda + q, \tag{5}$$

where a and q are parameters computed from training data.

To accurately reflect the R-D characteristics of the whole sequence, RC in H.265/HEVC uses the actual bit rate and λ values of the already coded LCUs and frames to update the model parameters, α and β, during the coding process. It is important to note that for intra-coded frames, α and β remain constant for all LCUs within the frame and are only updated after coding the whole frame [24, 25].

It is well-known that SC sequences differ from NC sequences in terms of their motion and texture. Namely, significant motion or abrupt changes are commonly found between consecutive frames of SC sequences. Moreover, many textures found in these sequences are computer-generated. To improve coding performance of SC sequences, a number of coding tools are incorporated in modern video codecs. For example, H.265/HEVC incorporates the SC coding (SCC) extensions, which include the following coding tools: Transform Skip (TS), Intra-Block Copy (IBC), Palette mode (PAL), Cross-Component Prediction (CCP), Adaptive Color Transforms, (ACTs), and Residual Differential Pulse Code Modulation (RDPCM) [26, 27]. The the third generation of Audio Video Standard (AVS3) incorporates string prediction (SP), also known as string matching, as an effective SCC tool. [28**?** –31]. TS bypasses the transform after intra-prediction to avoid spreading the energy associated with discontinuities in the residual signal over a wide frequency range [33]. IBC predicts the block using any previously coded and reconstructed region within the same frame, similar to motion estimation/compensation in inter-prediction. PAL enhances the prediction-then-transform representation of blocks that contain a limited number of different color values. CCP exploits the correlation among color components by predicting and scaling the residual of the second or third color component using as reference the residual of the first color component. ACTs remove inter-color component redundancy by adaptively converting the residual to different color spaces at the coding unit (CU) level. RDPCM predicts residual values in the horizontal or vertical direction using the immediately adjacent residual values. SP breaks the CU to be encoded into multiple strings of variable size and searches for the best-matching reference

strings within a previously coded and reconstructed region.

The motion and texture characteristics of SC sequences, unfortunately, usually result in RC methods performing poorly when applied to these sequences. Recently, several methods have been proposed to improve the performance of the R-λ model in H.265/HEVC for the coding of SC sequences. These methods account for the characteristics of these sequences and update the model parameters accordingly. For example, in [34], the authors propose to account for any significant motion and abrupt changes between consecutive frames to distribute the bit budget. Specifically, their method assigns a bit budget to each frame based on its complexity as measured by the prediction error within a sliding window. Once bit allocation is completed at the frame and LCU levels, λ is adjusted by a factor that reflects the consumption of bits. Although this method has been shown to outperform the current RC method used by H.265/HEVC, it still depends on accurately estimating the model parameters. The work in [35] improves the work in [34] by pre-computing the coding complexity of incoming frames after string them in a buffer. Based on this complexity analysis, the target bit rate for the current frame is estimated based on the buffer status and the complexity of the next few frames. At the LCU-level, the target bit rate is estimated based on the prediction error of motion estimation. This method, however, inevitably introduces delays as it requires analyzing incoming frames before making bit allocation decisions. In [36], the authors propose to first classify frames into key and non-key frames. Their method then uses two linear rate quantization (RQ) models to encode the classified frames. Frame classification and the linear RQ models are based on the Sum of Absolute Differences (SAD) between consecutive frames. Despite its advantages, this method does not include strategies for block-level OBA and QP estimation. More recently, the work in [37] proposes to classify Coding Tree Units (CTUs) into three categories: Text-CTUs (T-CTUs), Screen Image-CTUs, and Nature Image-CTUs. Three R-λ models with distinct parameters are then used to reflect the distinct R-D relationships of the three categories of CTUs. This strategy, which is similar to that used previously for High Dynamic Range content in [20], still relies on updating the parameters of the three R-λ models.

III. Proposed RC method

Since our contributions are at the frame- and block-levels, we follow the same uniform bit allocation strategy used by RC in H.265/HEVC at the GOP-level. By doing this, we can compare, in a fair manner, the performance of our RC method against the R-λ model of H.265/HEVC, as well as recently proposed RC methods for SC sequences based on H.265/HEVC. This particular bit allocation strategy uniformly distributes the total bit budget over the GOPs, with possible fluctuations related to the number of remaining uncoded GOPs and the remaining bit budget [1]. More specifically, at the GOP-level, the current GOP, G, is assigned T_G bits, which are determined based on the frame rate, F; the number of frames in the current GOP, N_G; the number of frames already coded in the current GOP, $\hat{N}_G$; the number of bits spent by the already coded GOPs, $\hat{T}_{1:G-1}$; and a smoothing window, S_w [1, 38]:

$$T_G = N_G \left(\frac{\frac{R_{target}}{F}(\hat{N}_G + S_w) - \hat{T}_{1:G-1}}{S_w} \right). \quad (6)$$

After allocating bits at the GOP-level, our RC method follows four main steps:

1) Bit Allocation at the frame-level.
2) Bit Allocation at the block-level.
3) Approximation of block-level R-D curves and λ.
4) Approximation of block-level R-QP curves and QPs.

It is important to note that the approximation of the block-level λ (Step 3) is necessary, as λ controls the trade-off between rate and distortion by adjusting the number of bits spent on blocks (see Eq. 2).

A. Bit allocation at the frame-level

Given T_G bits assigned to the current GOP, we assign T_f bits to the current frame, f, according to its coding cost, C_f, and the number of bits left for the uncoded frames in the GOP G, denoted by $\tilde{T}_G$:

$$T_f = \frac{\mathrm{C}_f}{\sum_{i=f}^{N_G} \mathrm{C}_i} \times \tilde{T}_G. \quad (7)$$

For the case of intra-predicted frames, the coding cost is denoted by C_f^{intra}. This cost only accounts for the texture of the frame, as intra-predicted frames do not depend on previously encoded frames. Specifically, we quantify the coding cost of an intra-predicted frame by the average of the Hadamard Transform values of its B constituent blocks:

$$\mathrm{C}_f^{intra} = \frac{\sum_{i=1}^{B} \mathrm{HAD}_i}{B}, \quad (8)$$

where HAD_i is the Hadamard Transform value of the i^{th} block, which is calculated as the sum of absolute differences between pixel values and their corresponding intra-predictions in the horizontal and vertical directions [39]. The Hadamard Transform is a fast and accurate way to estimate the coding cost of a block to be coded using angular intra-prediction and is currently employed in the reference implementation of the H.265/HEVC standard [38]. A low C_f^{intra} value indicates the majority of blocks in frame f have a smooth texture and are therefore relatively easy to code.

For the case of inter-coded frames, the coding cost is denoted by C_f^{inter}. This cost accounts for both the frame's texture and relative motion with respect to previously coded frames. Specifically, we quantify the coding cost of an inter-predicted frame by the average of the gradient values of the residual signals of its B constituent blocks:

$$\mathrm{C}_f^{inter} = \frac{\sum_{i=1}^{B} \mathrm{GRD}_i}{B}, \quad (9)$$

where GRD_i is the average of gradient values of the residual signal of the i^{th} block, computed in the horizontal, vertical

and diagonal directions, as follows:

$$\begin{aligned}\text{GRD}_i =& \frac{1}{n \times n} \sum_{r,c} |r^i_{r,c} - r^i_{r+1,c}| \\ &+ |r^i_{r,c} - r^i_{r,c+1}| + |r^i_{r,c} - r^i_{r+1,c+1}|,\end{aligned} \tag{10}$$

where $n \times n$ is the size of the i^{th} block and $r^i_{r,c}$ is its residual value at position (r, c). A low GRD value indicates an easy to code block as the absence of high gradient values indicates a smooth residual signal. Since C^{inter}_f is computed based on residual signals, it accounts for the relative motion of frame f with respect to previously coded frames.

B. Bit allocation at the block-level

Given T_f bits assigned to the current frame, we assign T_b bits to the current block, b, according to its coding cost, C_b, and the number of bits left for the uncoded blocks, denoted by $\tilde{T}_f$:

$$T_b = \frac{\text{C}_b}{\sum_{i=b}^{B} \text{C}_i} \times \tilde{T}_f. \tag{11}$$

We quantify the coding cost of an intra-predicted block by its Hadamard Transform, HAD_b. We quantify the coding cost of an inter-coded block by the average of gradient values of its residual signal, r_b, in the horizontal, vertical and diagonal directions, GRD_b, as computed in Eq. 10. The target bit rate for block b is then $R_b = \frac{T_b}{n \times n}$.

C. Approximation of block-level R-D curves and λ

The R-λ model in Eq. 4 has been shown to be very effective once a set of parameters, $\{\alpha, \beta\}$, is obtained from training data. For the particular case of NC sequences, the set of parameters currently used by the reference implementation of the H.265/HEVC standard allows attaining a target bit rate accurately with low computational complexity. However, such a set of parameters may not be effective for SC sequences, whose content is well-known to be considerably different from that of NC sequences. To illustrate this, let us take the R-D relationship of the NC sequence *ParkScene* and the SC sequence *ChineseEditing*, as shown in Fig. 1 (1^{st} row). The slope of the R-D curve of the NC sequence *ParkScene* can indeed be approximated by $\lambda = \alpha R^{\beta}_{bpp}$ with $\alpha = 6.38$ and $\beta = -1.62$, which are values very close to those used by H.265/HEVC for RC (i.e., $\alpha = 6.75$ and $\beta = -1.78$). The slope of the R-D curve of the SC sequence *ChineseEditing* can also be approximated by an R-λ model, but with parameters $\alpha = 2.40$ and $\beta = -3.02$, which greatly differ from those used by H.265/HEVC for RC.

Fig. 1 (2^{nd} row) shows how a piecewise linear function can also approximate the R-D curves in Fig. 1 (1^{st} row) by sampling the curves and interpolating linearly between adjacent R-D points. Specifically, one can select N control points where the i^{th} point, $p_i = (\hat{R}_i, D_i)$, represents the actual measured rate, $\hat{R}_i$, and actual distortion, D_i. Piecewise linear interpolation can then be used to estimate any rate and distortion point [40]. Although such an approach has been shown to reduce the complexity of estimating the R-D characteristics of video sequences without significantly reducing the RC performance, it still requires pre-computing N control points for the whole video sequence [41].

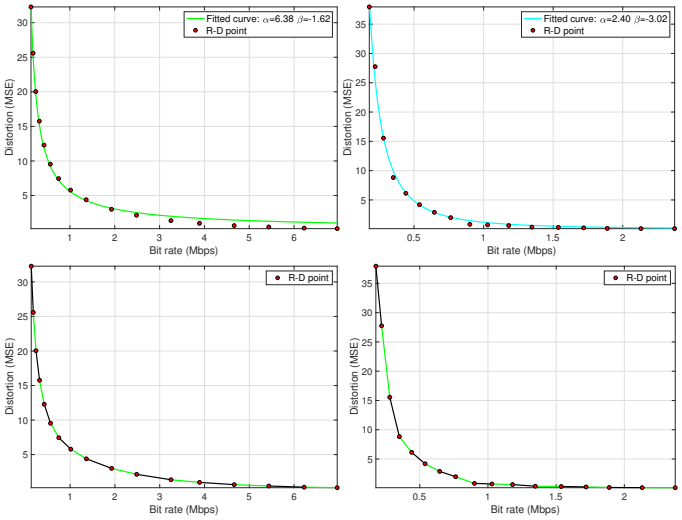

Fig. 1: (1^{st} row) The R-D relationship of (left) the NC sequence *ParkScene* and (right) the SC sequence *ChineseEditing*. Both sequences are coded using a GOP structure of I-P-P-P frames. (2^{nd} row) The R-D curves are approximated by a piecewise linear function by interpolating linearly between adjacent R-D points.

In this work, we leverage the low complexity of linear approximations and the availability of control points within the current frame to approximate the R-D characteristics of each block of the current frame. More specifically, we propose to linearly interpolate among $N \geq 2$ control points, each representing actual measured rate and distortion values of previously coded blocks within the same frame, whose characteristics are very similar to those of the current block. We use the coding cost of blocks, as defined in Section III-B, to quantitatively measure the similarity among blocks. We select $p_i = (\hat{R}_i, D_i)$ associated with the i^{th} already coded block as a control point for the b^{th} block, with $i < b$, if the following two criteria are met:

Criterion 1 - coding cost: $(1-\rho)\text{C}_b \leq \text{C}_i \leq (1+\rho)\text{C}_b$, where $\rho \ll 1$ is a small constant. Criterion 1 allows identifying those previously coded blocks whose coding costs are very similar to that of the b^{th} block, i.e., C_b. Similar coding costs mean similar R-D characteristics.

Criterion 2 - actual measured rate: $(1-\sigma)R_b \leq \hat{R}_i \leq (1+\sigma)R_b$, where $\sigma \ll 1$ is a small constant. Criterion 2 allows selecting those previously coded blocks that, apart from fulfilling Criterion 1, have been coded at a bit rate very similar to the target bit rate of the b^{th} block, i.e., R_b.

Fig. 2 exemplifies three sample collections of $N \geq 2$ control points for a P-frame of the SC sequence *Viking*. Note that as the number of control points increases, the variety of the actual measured rate and distortion values also increases, which in turn helps to more accurately represent the R-D characteristics of similar blocks. Also note that it is possible to fit a line using a least-squares criterion over the set of $N \geq 2$ control points

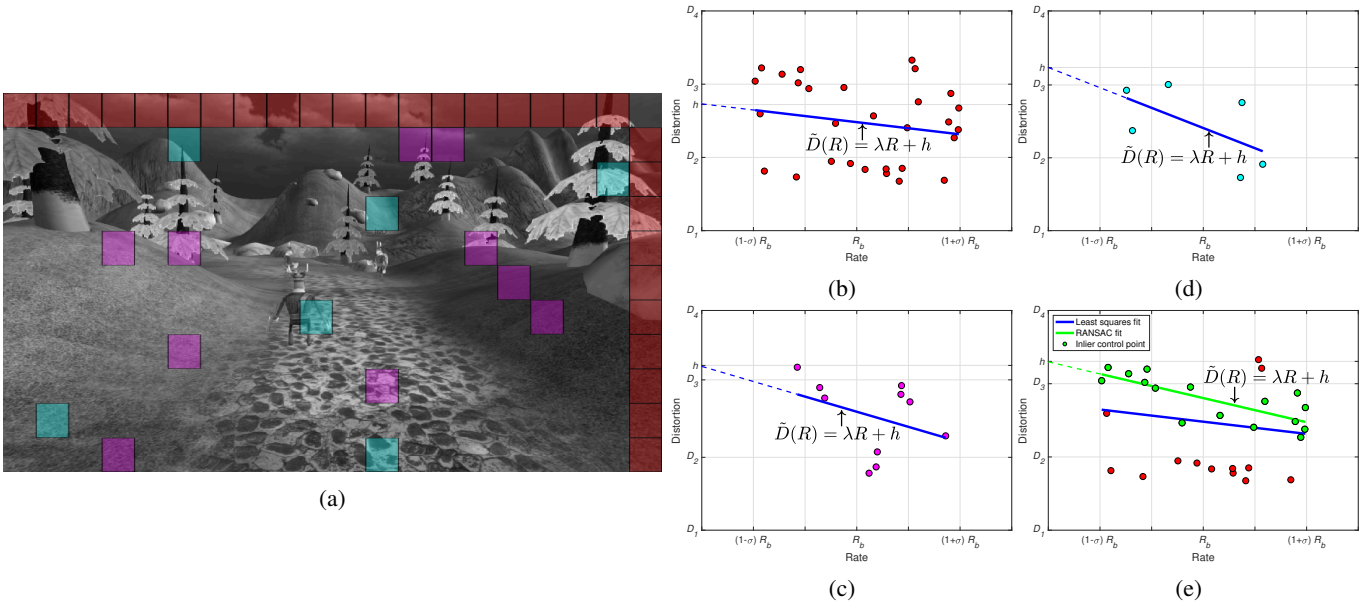

Fig. 2: (a) A P-frame of the SC sequence *Viking* coded using an I-P-P-P GOP structure. Similar blocks are highlighted in three distinct colors. The similarity is measured in terms of the coding cost of the residual blocks (see Eq. 10). R-D control points and the line fitted (least-squares criterion) to approximate the corresponding R-D relationship of (b) low-cost blocks (highlighted in red on the frame), (c) medium-cost blocks (highlighted in magenta on the frame), and (d) high-cost blocks (highlighted in blue on the frame). (e) The control points associated with the low-cost blocks (red blocks) are used to fit a line using RANSAC (in green).

to approximate a linear R-D relation, $\tilde{D}(R)$, of the form:

$$\tilde{D}(R) = \tilde{\lambda}R + h, \tag{12}$$

where $\tilde{\lambda}$ is the slope of the linear interval and h is the distortion intercept, i.e., the point at which the line crosses the distortion axis. The slope in Eq. 12 is a very good approximation of the slope of the R-D curve of the control points over a piecewise linear segment. In turn, this slope is also a very good approximation of λ, i.e., the slope of the R-D curve of the current block. The approximated slope, $\tilde{\lambda}$, can then be used to drive the coding process of the current block and compute a QP value using the linear QP-λ relation in Eq. 5. However, one must account for two important aspects. First, the set of $N \geq 2$ control points, although selected to resemble the R-D characteristics of the current block as close as possible, may include outliers that negatively affect the linear approximation in Eq. 12. Second, using the linear QP-λ relation in Eq. 5 implies relying on a model whose parameters, $\{a, q\}$, may not accurately reflect the QP-λ relationship of the current block. We address the first concern by introducing RANSAC into the linear approximation of Eq. 12, as explained next. We address the second concern in Section III-D, where we propose to compute the QP of the current block by also using linear interpolations.

RANSAC [42] randomly draws two control points from the set of $N \geq 2$ control points and computes a candidate linear R-D relation, $\tilde{D}(R)_{candidate}$, using Eq. 12. Control point p_i is an inlier of the line defined by $\tilde{D}(R)_{candidate}$ if and only if the squared distance from p_i to such line is less than a threshold, th. RANSAC tests $\tilde{D}(R)_{candidate}$ against all $N \geq 2$ control points to determine the number of inliers. After a given number of iterations, the candidate linear R-D relation with the most inliers is selected. This is illustrated in Fig. 2(e).

It is expected that for some blocks the number of control points found is $N < 2$, e.g., for the initial block of a frame. In such cases, we follow a low-complexity approach to approximate the R-D curve by using at most $N = 2$ control points representing actual measured rate and distortion values of the current block, as detailed in Algorithm 1.

Algorithm 1: Approximation of the R-D curve of block b by using actual measured rate and distortion values.

Require: $r_b, d_b, R_b, QP_{max}, QP_{min}$
1: $QP_1 \leftarrow \frac{\Delta QP}{db-1} \times r_{b_{depth}} + QP_{min} - \frac{\Delta QP}{db-1}$
2: $QP_1 \leftarrow \texttt{clip}(QP_1, QP_{min}, QP_{max})$
3: $(\hat{R}_1, D_1) \leftarrow \texttt{entropyCode}(r_b, QP_1)$
4: **if** $\hat{R}_1 > R_b$ **then**
5: $\quad QP_2 \leftarrow \lfloor QP_1 \times (1 + \frac{\hat{R}_1 - R_b}{R_b}) \rfloor$
6: **else**
7: $\quad QP_2 \leftarrow \lfloor QP_1 / |1 - \frac{\hat{R}_1 - R_b}{R_b}| \rfloor$
8: **end**
9: $QP_2 \leftarrow \texttt{clip}(QP_2, QP_{min}, QP_{max})$
10: $(\hat{R}_2, D_2) \leftarrow \texttt{entropyCode}(r_b, QP_2)$
11: $\tilde{\lambda} \leftarrow |\frac{D_1 - D_2}{\hat{R}_1 - \hat{R}_2}|$
12: **return** $\tilde{\lambda}$

Algorithm 1 requires as input the bit depth of block b in the pixel domain, d_b; its residual signal, r_b; its target bit rate, R_b; and the maximum and minimum QP values accepted by the encoder, QP_{max} and QP_{min}, respectively. Line 1 computes the QP associated with the first control point by using a linear relation between the range of QP values used by the encoder and d_b, where $\Delta QP = QP_{max} - QP_{min}$ and $r_{b_{depth}}$ denotes the bit depth of the residual signal. The higher $r_{b_{depth}}$,

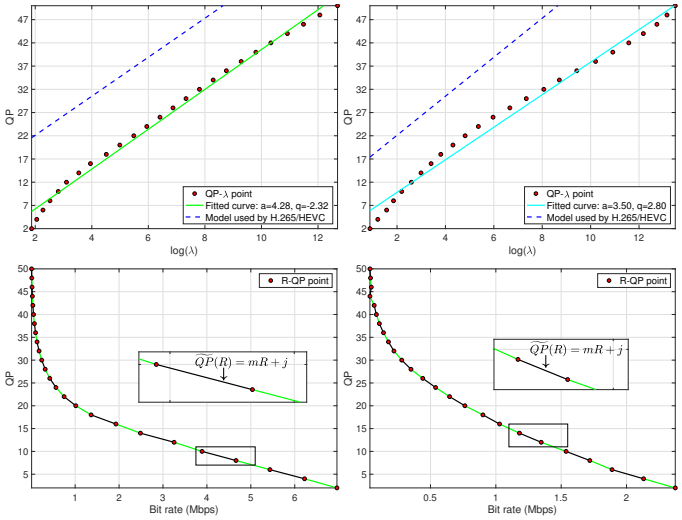

Fig. 3: (1^{st} row) The linear QP-λ relation of (left) the NC sequence *ParkScene* and (right) the SC sequence *ChineseEditing* encoded using a GOP structure of I-P-P-P frames. (2^{nd} row) The R-QP relation of each sequence is approximated by a piecewise linear function interpolating linearly between adjacent R-QP points.

the larger QP_1, which implies that high-bit depth residual signals require a relatively large QP for the bit rate to get close to R_b. Line 2 guarantees that QP_1 is within the range $[QP_{min}, QP_{max}]$. Line 3 entropy encodes r_b with QP_1 to compute the first control point $p_1 = (\hat{R}_1, D_1)$. Lines 4-8 compute the QP associated with the second control point by using a linear relation between rates $\{\hat{R}_1, R_b\}$ and QP_1. If $\hat{R}_1 > R_b$, then QP_2 must be $> QP_1$; otherwise, QP_2 must be $< QP_1$. Line 10 entropy encodes r_b with QP_2 to compute the second control point, $p_2 = (\hat{R}_2, D_2)$. The two control points are used to linearly interpolate $\tilde{\lambda}$ in line 11.

D. Approximation of block-level R-QP curves and QP

To compute the QP of the current block, we depart from the QP-λ model in Eq. 5 and instead approximate the R-QP relationship using RANSAC. Fig. 3 (1^{st} row) shows the linear QP-λ relations of the NC sequence *ParkScene* and the SC sequence *ChineseEditing* according to the model in Eq. 5. It is clear that this model poorly approximates these linear QP-λ relations. Fig. 3 (2^{nd} row) shows instead the actual R-QP curves of these sequences. Note that these curves are not linear, but can be approximated by a piecewise linear function interpolating linearly between adjacent R-QP points, where each linear segment is of the form:

$$\widetilde{QP}(R) = mR + j, \tag{13}$$

where m is the slope of the line and j is the QP intercept, i.e., the point at which the line crosses the QP axis. Based on this observation, we propose to use RANSAC with the same $N \geq 2$ control points used to approximate the R-D curve of the current block to compute its QP value according to Eq. 13. This is possible since each control point is associated with a QP value. Consequently, the i^{th} control point can also be expressed as $p_i = (\hat{R}_i, QP_i)$, where QP_i is the QP value that results in rate $\hat{R}_i$ with distortion D_i. For those blocks where the number of control points found is $N < 2$, we follow an equivalent low-complexity approach to that in Algorithm 1 to approximate the R-QP curve by using at most $N = 2$ control points, with the only difference that in line 11 the algorithm computes $\widetilde{QP} \leftarrow \left|\frac{QP_1 - QP_2}{\hat{R}_1 - \hat{R}_2}\right|$.

Our complete approach to computing the value of $\tilde{\lambda}$ and QP for block b is embodied in Algorithm 2. Apart from the inputs required by Algorithm 1, Algorithm 2 requires the costs of the already coded blocks in the current frame, $\{C_0, \ldots, C_{b-1}\}$, parameters ρ and σ (used by the two criteria explained in Section III.B), and set P, which stores each control point as $p_i = (\hat{R}_i, D_i, QP_i)$. In lines 1-5, Algorithm 2 selects the control points according to the two criteria. In line 7, Algorithm 2 uses RANSAC to compute $\tilde{\lambda}$ and $\widetilde{QP}$ if set P has at least two elements. If $|P| < 2$, function `approxLamdaQP` is used in line 9 to compute $\tilde{\lambda}$ and $\widetilde{QP}$ by using Algorithm 1. The algorithm returns $\{\tilde{\lambda}, QP\}$ in line 12 after clipping $\widetilde{QP}$ in a narrow range.

Algorithm 2: Computation of block-level λ and QP for block b.

Require: $\{C_0, \ldots, C_{b-1}\}, r_b, d_b, R_b, QP_{max}, QP_{min}, \rho, \sigma, P = \emptyset$

1: **for** $i = 1 \rightarrow b - 1$ **do**
2: **if** $(1-\rho)C_b \leq C_i \leq (1+\rho)C_b$ **AND** $(1-\sigma)R_b \leq \hat{R}_i \leq (1+\sigma)R_b$
3: $P \leftarrow P \cup \{(\hat{R}_i, D_i, QP_i)\}$
4: **end**
5: **end**
6: **if** $|P| \geq 2$
7: $\{\tilde{\lambda}, \widetilde{QP}\} \leftarrow$ `RANSAC`(P)
8: **else**
9: $\{\tilde{\lambda}, \widetilde{QP}\} \leftarrow$ `approxLambdaQP`$(r, d_b, R_b, QP_{max}, QP_{min})$
10: **end**
11: $QP \leftarrow$ `clip`$(\widetilde{QP}, QP_{min}, QP_{max})$
12: **return** $\{\tilde{\lambda}, QP\}$

IV. Performance Evaluation

We embed our RC method into a standard H.265/HEVC encoder with the Screen Content Coding (SCC) tools [26] (HM16.9+SCM-8.0 [43]). Our test data set comprises several 4:4:4-RGB, 4:4:4-YUV and 4:2:0-YUV SC sequences with 8-bit and 10-bit precision from the H.265/HEVC common test conditions for screen content coding (CTC-SCC) [44] and the verification test for screen content coding (VT-SCC) [45], as tabulated in Tables I-III. These test sequences cover a wide range of characteristics in terms of length, smoothness, scene complexity, type of content, and motion. We use all frames of these test sequences as specified in the CTC-SCC and the VT-SCC [44, 45]. We use the All-Intra (AI), Low Delay-B (LD-B), and Random Access (RA) coding configurations of the CTC-SCC and the VT-SCC with an LCU size of 64×64 and the SCC tools. To choose the target bit rates, we first encode each video sequence at the four fixed QPs used in the CTC-SCC and the VT-SCC to ensure a wide range of qualities for the compressed videos. Then, the actual bit rates used to compress the video sequences at these four fixed QPs are set

as the target bit rates for RC. These target bit rates, in Mbps, are tabulated in Tables I-III. Note that CTC-SCC specifies four QPs for all sequences in the set, i.e., 37, 32, 27, and 22; while VT-SCC specifies QPs that depend on the sequence. These QPs are also tabulated in Tables I-III.

We evaluate five RC methods: 1) The RC method of the H.265/HEVC reference implementation HM16.9+SCM-8.0 [43], which is based on a R-λ model, 2) a baseline RC method, 3) the RC method proposed in [35], 4) the RC method proposed in [37], and 5) our RC method. The RC method in [35] uses a similar approach to ours. Namely, it allocates bits to each frame and block according to their complexity as measured by their prediction error. This method, however, still relies on the R-λ model of H.265/HEVC. The method in [37] uses three R-λ models with distinct parameters to encode three different types of CTUs. The baseline RC method computes the λ and QP values of the b^{th} LCU by using a recursive algorithm that guarantees to attain its target bit rate, R_b, with an accuracy of $1\pm\xi$, where $\xi \ll 1$ and an accuracy $= 1$ means that its measured bit rate, $\hat{R}_b$, is equal to R_b. The steps of this recursive algorithm are listed in Algorithm 3.

Algorithm 3: Computation of λ and QP for LCU b using recursion.

Require: $r_b, R_b, \{\alpha, \beta, a, q\}, QP_{max}, QP_{min}, \xi$
1: $max_{QP} \leftarrow QP_{max}$; $min_{QP} \leftarrow QP_{min}$
2: $QP \leftarrow \lfloor \frac{QP_{max}}{2} \rfloor$
3: $QP \leftarrow \texttt{clip}(QP, QP_{min}, QP_{max})$
4: $\lambda = \exp(QP - q)/a$
5: $\hat{R}_b \leftarrow \texttt{entropyCoding}(r_b, QP)$
6: **while** $\hat{R}_b \notin [(1-\xi)R_b, (1+\xi)R_b]$ **OR** $|max_{QP} - min_{QP}| > 1$
7: **if** $\hat{R}_b > R_b$ **then**
8: $min_{QP} \leftarrow QP$
9: $diff \leftarrow \lfloor | \frac{QP - max_{QP}}{2} | \rfloor$
10: $QP \leftarrow (QP + diff)$
11: **elseif** $\hat{R}_b < R_b$ **then**
12: $max_{QP} \leftarrow QP$
13: $diff \leftarrow \lfloor | \frac{QP - min_{QP}}{2} | \rfloor$
14: $QP \leftarrow (QP - diff)$
15: **else**
16: **return** $\{\lambda, QP\}$
17: **break**
18: **end**
19: $QP \leftarrow \texttt{clip}(QP, QP_{min}, QP_{max})$
20: $\lambda = \exp(QP - q)/a$
21: $\hat{R} \leftarrow \texttt{entropyCoding}(r_b, QP)$
22: **end**
23: **return** $\{\lambda, QP\}$

Algorithm 3 entropy encodes LCU b with an initial QP computed according to the maximum QP value accepted by the encoder (see lines 2, 3 and 5). The initial λ value is computed using the linear model in Eq. 5 (see line 4). In lines 6-22, the algorithm recursively entropy encodes LCU b as long as the actual measured rate, $\hat{R}_b$, is outside the range $[(1-\xi)R_b, (1+\xi)R_b]$ or the absolute difference between max_{QP} and min_{QP} is greater than 1. Each time LCU b is entropy encoded with a new QP, the actual measured rate, $\hat{R}_b$, is compared against R_b. If $\hat{R}_b > R_b$, the value of min_{QP} is updated to indicate that the minimum QP value used thus far is QP (see line 8). QP is then increased in line 10 by $diff$, which is computed based on max_{QP} and QP (see line 9). If $\hat{R}_b < R_b$, the value of max_{QP} is updated to indicate that the maximum QP value used thus far is QP (see line 12). QP is then decreased in line 14 by $diff$. Note that the value of $diff$ decreases as the iteration number increases. This guarantees that the range of possible QP values, i.e., $[max_{QP}, min_{QP}]$, to entropy encode LCU b is iteratively reduced until the most appropriate QP value is found. If $\hat{R}_b = R_b$, the algorithm returns the current λ and QP values and breaks the *while* loop (see lines 16 and 17). In line 23, the algorithm returns the final λ and QP values. To have a fair comparison, the baseline method is also embedded into the same standard H.265/HEVC encoder (HM16.9+SCM-8.0 [43]) and follows the same GOP-level and frame-level bit allocation as that followed by the RC method of H.265/HEVC.

Since RC methods are only used during the encoding process to select the appropriate set of QPs, the decoding process is not affected. Hence, we use the standard H.265/HEVC decoder HM16.9+SCM-8.0 [43] to reconstruct all sequences compressed by all evaluated RC methods. We measure the RC accuracy in terms of the bit-rate error (BRE) between the target bit rate, R_{target}, and the attained bit rate, $\hat{R}$:

$$BRE = \frac{\hat{R} - R_{target}}{R_{target}} \times 100. \tag{14}$$

The four different target bit rates evaluated for each test sequence allow computing a Bjontegaard-based metric for BRE, PSNR and bit rate (BR) values. To compute BD-BRE values, we use absolute BRE values to measure any improvements to the RC accuracy. A negative BD-BRE value then indicates an increase in RC accuracy; a positive BD-PSNR value indicates a PSNR gain at the same bit rate; and a negative BD-BR value indicates a bit rate reduction at the same PSNR quality.

Tables IV, V, and VI tabulate the BD-BRE, BD-PSNR and BD-BR values of our RC method and the methods in [35] and [37] for the AI, LD-B and RA configurations, respectively, when compared against the RC method in H.265/HEVC and the baseline RC method. Under the AI configuration, our RC method particularly outperforms the methods in [35] and [37] for 4:2:0-YUV sequences. Note that although the method in [37] performs, on average, strongly in terms of RC accuracy for 4:4:4-RGB sequences (see average BD-BRE values), its performance is not very strong in terms of PSNR gains (see average BD-PSNR value) and bit rate reductions (see average BD-BR values). This is mainly due to the fact this method, similarly to the one in [35], still relies on an R-λ model, whose performance depends on parameters $\{\alpha, \beta\}$. Moreover, both methods, like the RC method used by the H.265/HEVC reference implementation, do not have a parameter refining process for parameters $\{a, q\}$, which are used to compute QP values based on λ (see Eq. 5). The lack of such a refining process further contributes to their lower performance compared to our method. For the three types of sequences, our method not only attains, on average, the bit rate more accurately than the methods in [35] and [37], but also provides

TABLE I: Characteristics of the 4:4:4-RGB test sequences and target bit rates.

Sequence	Resolution	No. frames	fps	$R_{target}(Mbps)$											
				AI				LD-B				RA			
			QP	22	27	32	37	22	27	32	37	22	27	32	37
VenueVu	1920×1080	300	30	118.29	73.01	45.30	24.73	20.95	10.00	5.13	2.73	17.23	8.24	4.33	2.20
FlyingGraphics		600	60	61.30	45.41	34.09	23.52	32.55	19.57	11.82	7.27	29.32	17.65	10.39	6.31
Desktop		600	60	37.68	33.71	29.24	24.28	0.93	0.89	0.83	0.76	1.50	1.38	1.25	1.11
Console		600	60	30.71	25.74	21.97	17.13	4.51	3.63	2.85	2.16	4.72	3.71	2.92	2.21
MissionControlClip3		600	60	96.65	67.34	45.36	29.71	3.85	2.10	1.10	0.59	4.94	2.82	1.66	1.00
EBURainFruits*		250	50	136.34	81.97	47.49	26.48	27.88	11.09	4.73	2.14	21.12	8.88	4.21	2.11
Kimono1*		120	24	104.55	38.43	11.59	6.19	51.62	9.62	3.00	1.35	37.06	8.31	2.53	1.13
Map	1280×720	600	60	39.65	27.08	17.28	10.71	1.54	0.99	0.60	0.36	2.02	1.32	0.82	0.50
Robot		300	30	36.77	17.45	7.51	3.63	5.49	1.80	0.62	0.24	5.20	1.94	0.74	0.32
Viking		300	30	33.03	19.18	9.89	4.99	4.44	1.89	0.81	0.37	4.00	1.85	0.83	0.39
WebBrowsing		300	30	6.33	5.09	3.82	2.68	0.16	0.13	0.10	0.08	0.33	0.26	0.20	0.15
Programming		600	60	36.78	24.86	16.83	11.16	6.02	3.33	1.62	0.74	5.48	2.90	1.47	0.75
SlideShow		500	20	7.11	4.48	2.75	1.65	1.01	0.60	0.34	0.19	1.16	0.68	0.39	0.23
			QP	20	28	34	38	20	28	34	38	20	28	34	38
KristenAndSaraScreen	1920×1080	480	60	105.28	60.81	38.97	28.19	3.72	1.30	0.62	0.39	4.96	2.02	1.15	0.79
			QP	24	26	28	30	24	26	28	30	24	28	30	34
BigBuckBunnyStudio	1920×1080	400	50	56.91	49.37	42.81	36.33	2.14	1.75	1.44	1.13	3.05	2.08	1.69	1.17
			QP	34	36	38	40	34	36	38	40	34	36	38	40
ClearTypeSpreadsheet	1920×1080	240	30	16.07	14.82	13.21	11.71	0.15	0.14	0.13	0.11	0.66	0.60	0.54	0.48
EnglishDocumentEditing		240	30	25.14	22.91	19.89	16.83	0.29	0.26	0.22	0.18	1.03	0.95	0.83	0.72
			QP	28	32	34	36	28	32	36	40	28	34	38	42
CircuitLayoutPresentation	1920×1080	240	30	30.39	28.12	26.90	25.52	3.35	2.32	1.43	0.75	3.99	2.47	1.56	1.01
ChineseDocumentEditing		240	30	51.98	46.90	43.89	40.72	0.71	0.59	0.46	0.32	2.19	1.84	1.52	1.10

*10-bit sequence.

TABLE II: Characteristics of the 4:4:4-YUV test sequences and target bit rates.

Sequence	Resolution	No. frames	fps	$R_{target}(Mbps)$											
				AI				LD-B				RA			
			QP	22	27	32	37	22	27	32	37	22	27	32	37
FlyingGraphics	1920×1080	600	60	78.29	62.01	46.23	32.92	20.95	10.00	5.13	2.73	41.19	25.24	15.30	8.86
Desktop		600	60	37.68	33.71	29.24	24.28	0.90	0.84	0.78	0.70	1.41	1.28	1.15	0.99
Console		600	60	34.86	30.31	25.56	20.76	5.26	4.43	3.56	2.75	5.53	4.50	3.56	2.72
MissionControlClip3		600	60	66.23	45.27	30.61	20.07	2.07	1.13	0.63	0.38	2.91	1.73	1.07	0.68
EBURainFruits*		250	50	87.16	51.72	29.58	16.28	12.69	5.41	2.53	1.22	10.43	4.89	2.48	1.28
Kimono1*		120	24	45.79	14.09	7.47	4.01	14.64	3.98	1.77	0.83	12.70	3.48	1.52	0.72
Map	1280×720	600	60	54.63	40.50	27.53	16.97	2.39	1.59	0.99	0.58	2.94	1.96	1.25	0.75
Robot		300	30	63.99	36.67	16.93	7.40	12.16	4.59	1.49	0.51	10.81	4.39	1.63	0.62
WebBrowsing		300	30	8.57	6.74	5.17	3.72	0.22	0.17	0.13	0.10	0.43	0.33	0.26	0.19
Programming		600	60	51.21	36.66	24.66	15.95	9.08	5.46	2.87	1.25	8.52	4.93	2.56	1.18
SlideShow		500	20	4.34	2.74	1.71	1.09	0.60	0.34	0.20	0.12	0.69	0.41	0.25	0.16
			QP	24	28	30	34	28	30	34	36	28	30	34	36
KristenAndSaraScreen	1920×1080	480	60	55.84	42.01	36.56	27.10	0.69	0.56	0.39	0.32	1.28	1.08	0.78	0.66
			QP	24	26	28	30	24	26	28	30	24	28	30	34
BigBuckBunnyStudio	1920×1080	400	50	39.02	33.39	28.50	24.06	1.25	1.02	0.83	0.65	1.92	1.32	1.08	0.73
			QP	30	34	36	38	30	34	36	38	30	34	36	38
ClearTypeSpreadsheet	1920×1080	240	30	16.46	13.83	12.33	10.95	0.15	0.13	0.12	0.11	0.67	0.57	0.52	0.46
EnglishDocumentEditing		240	30	26.86	21.13	18.69	15.74	0.30	0.23	0.20	0.17	1.10	0.89	0.79	0.70
			QP	28	32	34	36	28	32	36	40	28	34	38	42
CircuitLayoutPresentation	1920×1080	240	30	28.17	25.52	23.80	21.73	2.35	1.43	0.85	0.52	3.04	1.68	1.16	0.81
ChineseDocumentEditing		240	30	25.52	41.58	37.54	33.63	0.60	0.48	0.36	0.23	2.01	1.59	1.26	0.82

*10-bit sequence.

important PSNR gains and bit rate reductions. A similar trend is observed for the LD-B and RA configurations.

The most important reductions in BRE values are attained by our method for the 4:4:4-RGB sequence *Map* under the RA configuration (BD-BRE=-36.23%). This is a particularly challenging sequence, as it depicts numerous edges with computer generated textures, zooming and panning.

As expected, the baseline RC method attains the best performance in terms of RC accuracy, as it is based on an exhaustive search that finds the λ and QP values that attain the target bit rate of each LCU very accurately. For all test sequences, the baseline method attains, on average, absolute BRE values of only $\{0.010\%, 0.010\%, 0.011\%\}$ for the AI, LD-B and RA configurations, respectively. Such high RC accuracy, however, comes at the expense of high computational complexity, as each LCU may be entropy encoded several times due to the recursion (see Algorithm 3). The baseline RC method, however, may not attain the best reconstruction quality in

TABLE III: Characteristics of the 4:2:0-YUV test sequences and target bit rates.

Sequence	Resolution	No. frames	fps	$R_{target}(Mbps)$											
				AI				LD-B				RA			
			QP	22	27	32	37	22	27	32	37	22	27	32	37
MissionControlClip2	2560×1440	600	60	83.82	54.44	35.20	22.95	5.59	3.14	1.71	0.94	5.67	3.27	1.86	1.11
BasketBallScreen		600	60	95.54	65.43	46.91	32.99	3.99	2.04	1.12	0.67	4.94	2.79	1.69	1.08
FlyingGraphics	1920×1080	600	60	58.15	43.03	31.80	23.25	30.84	17.62	10.30	6.33	26.58	15.46	9.07	5.57
Desktop		600	60	35.69	28.29	22.32	18.80	1.46	1.13	0.95	0.84	1.93	1.53	1.26	1.05
Console		600	60	32.48	26.26	21.11	16.40-	4.58	3.59	2.77	2.15	4.82	3.69	2.80	2.15
MissionControlClip3		600	60	55.65	38.88	26.68	17.92	1.89	1.04	0.59	0.36	2.45	1.49	0.95	0.62
Map	1280×720	600	60	36.52	24.95	15.86	10.05	1.43	0.91	0.55	0.33	1.83	1.19	0.74	0.45
Robot		300	30	27.78	13.96	6.75	3.37	4.11	1.44	0.53	0.22	3.88	1.51	0.62	0.29
WebBrowsing		300	30	6.39	4.60	3.45	2.64	0.16	0.12	0.09	0.07	0.33	0.24	0.18	0.14
Programming		600	60	25.67	18.50	13.69	10.10	3.82	2.14	1.10	0.57	3.59	1.99	1.06	0.61
SlideShow		500	20	4.06	2.52	1.73	1.12	0.62	0.36	0.21	0.13	0.37	0.23	0.16	0.11
ChinaSpeed	1024×768	500	30	19.20	12.14	8.09	5.09	5.14	2.47	1.21	0.60	4.71	2.50	1.20	0.62
BasketBallDrillText	832×480	500	50	12.98	8.01	5.60	3.77	3.67	1.75	0.85	0.45	3.33	1.62	0.84	0.46
			QP	24	28	30	34	28	30	34	36	28	30	34	36
KristenAndSaraScreen	1920×1080	480	60	48.15	36.47	32.18	24.08	0.65	0.54	0.37	0.31	1.14	0.98	0.71	0.61
			QP	24	26	28	30	24	26	28	30	24	28	30	34
BigBuckBunnyStudio	1920×1080	400	50	34.87	29.90	25.70	22.01	1.10	0.89	0.72	0.58	1.69	1.17	0.98	0.67
			QP	24	30	34	36	28	30	34	36	28	34	38	42
ClearTypeSpreadsheet	1920×1080	240	30	18.99	13.53	11.39	10.66	0.15	0.14	0.11	0.11	0.63	0.46	0.40	0.34
EnglishDocumentEditing		240	30	25.62	18.50	14.62	13.23	0.26	0.23	0.18	0.16	1.10	0.63	0.51	0.39
			QP	28	32	34	36	28	32	36	40	28	34	38	42
CircuitLayoutPresentation	1920×1080	240	30	28.74	23.34	20.74	18.63	1.86	1.19	0.75	0.48	2.64	1.48	1.04	0.73
ChineseDocumentEditing		240	30	35.10	28.70	25.59	23.17	0.44	0.35	0.26	0.19	1.52	1.10	0.88	0.67

terms of PSNR values, as it relies on a uniform bit allocation strategy at the frame-level. This strategy, which is the same as the one followed by RC in H.264/HEVC, tends to assign the same number of bits to all frames with possible fluctuations related to the number of remaining uncoded frames and the remaining bit budget [21, 38]. Such a strategy does not account for the characteristics of the frames. Indeed, for the 4:2:0-YUV sequence *SlideShow* and the 4:4:4-RGB sequence *Kimono1* under the RA configuration, our RC method attains PSNR gains compared to the baseline method, as well as improvements in terms of BR-BRE and BR-BR values (see underlined values in Table VI). This shows that our RC method can more effectively distribute the bit budget among frames to attain the minimum distortion possible.

Figures 4 and 5 show PSNR and BRE values on a per-frame basis for the 4:4:4-RGB sequence *Map* and the 4:2:0-YUV sequence *MissionControlClip2* under the RA configuration for the RC method in H.265/HEVC, the RC methods in [35] and [37], and the proposed RC method. Per-frame BRE values are computed as the difference between the bits spent on each frame by the evaluated method and the bits spent by the encoder on the same frame when the corresponding fixed QP is used. Let us recall that the RC method in H.265/HEVC assigns a bit budget to each frame according to a uniform bit allocation strategy. This strategy may force the RC method to dramatically decrease or increase the number of bits assigned to some of the frames to guarantee that R_{target} is attained as accurate as possible for the whole sequence. Indeed, for the sequences plotted in Figs. 4 and 5, per-frame BRE values attained by the RC method in H.265/HEVC tend to be very large and vary significantly. This indicates that the parameter refining process of this RC method may fail to approximate the slope of the R-D curve of these SC sequences. Let us recall that such a refining process slowly adjusts the value of $\{\alpha, \beta\}$ as LCUs and frames are coded. However, it has been shown that if the initial $\{\alpha, \beta\}$ values significantly differ from the true ones, the RC accuracy tends to be low [46]. For the methods in [35] and [37], such variations on per-frame BRE values are considerably reduced. Also, note that the per-frame reconstruction quality (PSNR values), in general, tends to increase/decrease as per-frame BER values increase/decrease. Hence, if per-frame BER values fluctuate, per-frame PSNR values are also expected to fluctuate. Our RC method attains per-frame BRE values much closer to zero, which results in a more uniform distribution of PSNR values. Lower BRE values also result in a more constant bit rate, which is advantageous when the decoder relies on a constraint-size buffer to decode the bit stream [47]. This confirms that the proposed frame- and block-level bit allocation strategies, in conjunction with our approximation of the block-level R-D and R-QP curves, help to increase the RC accuracy significantly.

Figure 6 depicts subjective quality results for the SC sequences in Figs. 4 and 5. Specifically, this Fig. shows the reconstructed Y-component of two frames for which the RC methods in [35] and [37], and the proposed RC method attain a BER$= 0$, i.e., these methods spent the same number of bits on these frames as the number of bits spent when the corresponding fixed QP is used. The RC method of H.265/HEVC attains a BER$= \{6.45, -1.75\}$ for the *Map* and *MissionControlClip2* frames, respectively. Note that the negative effects of the large per-frame BRE values attained by the RC method of H.265/HEVC are visually evident in the reconstructed frames encoded by this method (see 1^{st} row of Fig. 6). Specifically, many of the edges and text within the map region of the *Map* frame, as well as most of the text and edges within the plot region of the *MissionControlClip2* frame

TABLE IV: BD-BRE, BD-PSNR and BD-BR values of test sequences under the AI configuration.

Sequence	Proposed RC method						RC method in [35]						RC method in [37]					
	vs. R-λ (HEVC)			vs. Baseline			vs. R-λ (HEVC)			vs. Baseline			vs. R-λ (HEVC)			vs. Baseline		
	BD-BRE	BD-PSNR	BD-BR	BD-BRE	BD-PSNR	BD-BR	BD-BRE	BD-PSNR	BD-BR	BD-BRE	BD-PSNR	BD-BR	BD-BRE	BD-PSNR	BD-BR	BD-BRE	BD-PSNR	BD-BR
4:4:4-RGB sequences																		
VenueVu	**-0.08**	**0.81**	**-13.30**	0.09	-0.13	0.05	0.00	0.30	-4.67	0.02	-0.16	0.04	-0.03	0.31	-5.22	0.01	-0.14	0.05
FlyingGraphics	**-4.17**	**2.02**	**-13.86**	0.02	0.00	0.05	-3.97	0.21	-1.87	0.30	-0.03	0.06	-4.05	0.89	-6.13	0.03	0.00	0.05
Desktop	**-2.67**	**0.78**	**-1.97**	0.08	-0.02	0.24	-1.67	0.64	-1.59	0.17	-0.28	0.33	-1.91	0.73	-1.86	0.14	-0.16	0.30
Console	**-1.98**	**2.26**	**-8.20**	0.08	-0.23	0.74	-1.02	1.13	-3.77	0.08	-0.33	0.89	-1.37	1.27	-5.32	0.07	-0.26	0.98
MissionControlClip3	**-3.97**	**0.63**	-5.58	0.32	-0.23	0.47	-3.58	0.60	-5.79	0.45	-0.31	0.54	-3.69	0.59	**-5.86**	0.42	-0.29	0.49
EBURainFruits*	**-1.57**	**0.29**	**-5.50**	0.27	-0.05	0.28	-1.32	0.25	-4.28	0.33	-0.11	0.36	-1.43	0.27	-4.89	0.31	-0.12	0.27
Kimono1*	**-0.33**	0.19	**-10.79**	0.30	-0.01	0.11	-0.23	0.10	-7.01	0.35	-0.05	0.25	-0.27	**0.22**	-5.22	0.33	-0.06	0.18
Map	**-0.20**	**0.92**	**-10.43**	0.59	-0.02	0.08	-0.08	0.15	-1.76	0.71	-0.03	0.12	**-0.20**	0.27	-2.39	0.88	-0.03	0.09
Robot	-0.42	**1.65**	**-35.66**	0.39	-0.01	0.10	-0.38	0.53	-15.44	0.67	-0.15	0.13	**-0.44**	0.83	-21.43	0.58	-0.15	0.11
Viking	**-11.23**	**0.43**	**-6.57**	0.63	-0.07	0.03	-9.90	0.31	-5.04	0.56	-0.09	0.03	-11.13	0.32	-5.36	0.63	-0.08	0.03
WebBrowsing	**-2.38**	**1.92**	**-9.57**	0.01	-0.12	0.51	-1.88	1.11	-5.53	0.15	-0.48	0.96	-2.10	1.85	-9.24	0.01	-0.13	0.59
Programming	**-2.54**	**1.17**	**-12.39**	0.14	-0.33	0.06	-1.85	0.61	-6.99	0.18	-0.02	0.05	-1.97	1.04	-10.52	0.14	-0.20	0.04
SlideShow	**-3.22**	**0.68**	-8.13	0.07	-0.03	0.11	-2.62	0.66	**-8.65**	0.11	-0.08	0.19	-2.96	0.63	-7.90	0.07	-0.16	0.18
KristenAndSaraScreen	**-2.22**	**0.81**	**-6.61**	0.37	-0.19	0.55	-1.69	0.64	-5.68	0.52	-0.28	0.78	-1.81	0.63	-5.52	0.41	-0.26	0.72
BigBuckBunnyStudio	**-9.66**	**0.71**	**-5.84**	0.72	-0.12	0.21	-8.05	0.23	-1.66	0.69	-0.18	0.27	-9.19	0.40	-3.53	0.64	-0.17	0.29
ClearTypeSpreadsheet	**-3.47**	**0.74**	**-2.30**	0.10	-0.43	0.22	-1.87	0.60	-1.68	0.17	-0.47	0.31	-3.22	0.62	-1.31	0.15	-0.42	0.28
EnglishDocumentEditing	**-1.34**	**1.08**	**-4.07**	0.21	-0.47	0.16	-0.49	0.38	-1.58	0.36	-0.54	0.24	-0.51	0.87	-3.63	0.32	-0.49	0.21
CircuitLayoutPresentation	-2.96	**1.19**	**-1.93**	0.15	-0.54	0.23	-0.88	0.59	-0.40	0.18	-0.65	0.31	**-2.99**	0.94	-1.29	0.17	-0.62	0.29
ChineseDocumentEditing	**-1.92**	**0.71**	**-1.91**	0.42	-0.14	0.48	-0.90	0.50	-1.21	0.78	-0.15	0.66	-1.08	0.51	-0.62	0.77	-0.16	0.65
4:4:4-RGB Average	**-3.12**	**0.96**	**-7.51**	0.30	-0.18	0.27	-2.34	0.49	-3.94	0.41	-0.32	0.37	-2.76	0.69	-4.90	0.37	-0.26	0.33
4:4:4-YUV sequences																		
FlyingGraphics	**-4.26**	**1.79**	**-10.55**	0.03	-0.02	0.06	-3.72	0.29	-1.88	0.39	-0.23	0.08	-3.88	0.94	-5.32	0.04	-0.03	0.07
Desktop	**-2.76**	**0.94**	**-2.78**	0.09	-0.01	0.22	-2.16	0.67	-1.90	0.13	-0.23	0.29	-2.10	0.79	-2.34	0.12	-0.06	0.21
Console	**-2.30**	**2.23**	**-7.01**	0.07	-0.27	0.82	-1.57	1.06	-3.08	0.09	-0.40	0.99	-1.98	1.49	-4.89	0.08	-0.31	1.03
MissionControlClip3	**-3.42**	**0.68**	-6.88	0.26	-0.13	0.22	-2.91	0.65	-7.01	0.35	-0.28	0.33	-2.96	0.62	**-6.95**	0.30	-0.22	0.31
EBURainFruits*	**-1.67**	**0.32**	**-7.22**	0.36	-0.11	0.29	-1.34	0.27	-5.48	0.42	-0.12	0.44	-1.55	0.29	-6.24	0.39	-0.11	0.35
Kimono1*	**-0.30**	**0.24**	**-11.38**	0.27	-0.02	0.15	-0.19	0.17	-7.60	0.41	-0.09	0.22	-0.19	0.18	-8.74	0.38	-0.11	0.20
Map	**-0.32**	**0.74**	**-6.98**	0.53	-0.03	0.12	-0.17	0.13	-1.25	0.64	-0.06	0.18	-0.28	0.26	-1.70	0.79	-0.05	0.13
Robot	-0.39	**1.41**	**-25.78**	0.43	-0.03	0.16	-0.26	0.46	-9.65	0.74	-0.91	0.21	**-0.40**	0.63	-13.44	0.64	-0.90	0.17
WebBrowsing	**-3.77**	**1.96**	**-8.86**	0.84	-0.23	0.63	-2.96	1.04	-4.94	1.02	-0.89	1.19	-3.05	1.85	-8.40	1.02	-0.32	0.83
Programming	**-3.57**	**1.10**	**-9.04**	0.12	-0.39	0.12	-2.59	0.55	-4.87	0.15	-0.21	0.11	-2.70	0.89	-7.05	0.12	-0.26	0.15
SlideShow	**-2.43**	**0.77**	**-9.95**	0.03	-0.08	0.09	-1.99	0.72	-10.68	0.05	-0.11	0.12	-2.19	0.73	-9.62	0.04	-0.13	0.11
KristenAndSaraScreen	**-2.67**	**0.65**	**-6.04**	0.31	-0.22	0.73	-1.89	0.63	-6.18	0.49	-0.37	0.82	-2.31	0.63	-5.98	0.47	-0.33	0.79
BigBuckBunnyStudio	**-9.15**	**0.60**	**-5.76**	0.81	-0.15	0.26	-7.27	0.20	-1.58	0.92	-0.21	0.34	-8.08	0.50	-5.06	0.85	-0.19	0.37
ClearTypeSpreadsheet	**-3.91**	**0.87**	**-3.23**	0.11	-0.37	0.19	-1.53	0.67	-2.92	0.14	-0.53	0.39	-2.06	0.83	-3.06	0.14	-0.44	0.31
EnglishDocumentEditing	**-2.40**	**1.17**	**-10.28**	0.18	-0.34	0.21	-1.59	0.60	-2.96	0.23	-0.42	0.27	-1.96	1.12	-5.52	0.21	-0.39	0.22
CircuitLayoutPresentation	**-5.12**	**1.24**	**-3.04**	0.16	-0.61	0.18	-3.82	0.60	-2.98	0.20	-0.67	0.22	-4.59	0.89	-2.28	0.19	-0.64	0.21
ChineseDocumentEditing	**-2.85**	**0.80**	**-3.66**	0.45	-0.12	0.53	-2.07	0.50	-1.66	0.87	-0.17	0.64	-2.32	0.46	-1.04	0.73	-0.14	0.67
4:4:4-YUV Average	**-2.84**	**1.07**	**-9.43**	0.25	-0.17	0.27	-2.11	0.55	-5.08	0.37	-0.25	0.37	-2.38	0.78	-6.58	0.32	-0.21	0.33
4:2:0-YUV sequences																		
MissionControlClip2	**-1.96**	**1.01**	**-14.00**	0.46	-0.03	0.68	-1.07	0.80	-11.81	0.62	-0.09	0.62	-1.37	0.69	-10.39	0.50	-0.07	0.55
BasketBallScreen	**-2.72**	**0.81**	**-8.71**	0.26	-0.07	0.43	-1.25	0.49	-4.90	0.43	-0.12	0.46	-1.90	0.60	-6.28	0.37	-0.11	0.45
FlyingGraphics	**-3.04**	**1.95**	**-14.81**	0.03	-0.05	0.07	-2.43	0.24	-2.15	0.31	-0.08	0.09	-2.45	0.84	-6.45	0.27	-0.04	0.10
Desktop	**-1.97**	**0.81**	**-4.30**	0.10	-0.29	0.32	-1.29	0.61	-3.41	0.09	-0.44	0.27	-1.41	0.69	-3.98	0.07	-0.36	0.24
Console	**-3.20**	**2.06**	**-23.81**	0.04	-0.13	0.34	-2.54	0.71	-8.97	0.05	-0.21	0.69	-1.41	0.69	-3.98	0.12	-0.31	0.73
MissionControlClip3	**-2.82**	**0.67**	-7.72	0.13	-0.17	0.55	-1.89	**0.67**	-7.83	0.13	-0.33	0.43	-1.92	0.66	**-7.89**	0.14	-0.19	0.35
Map	**-3.54**	**0.88**	**-10.57**	0.42	-0.01	0.10	-3.17	0.15	-2.50	0.79	-0.08	0.32	-3.14	0.28	-3.08	0.72	-0.01	0.23
Robot	**-1.88**	**1.63**	**-25.47**	0.32	-0.05	0.21	-1.56	0.53	-15.20	0.58	-0.20	0.11	-1.54	0.82	-18.86	0.62	-0.18	0.10
WebBrowsing	**-4.18**	**1.87**	**-15.75**	0.04	-0.23	0.43	-3.42	1.06	-9.02	0.16	-0.52	0.75	-3.56	1.75	-14.86	0.05	-0.67	0.83
Programming	**-3.50**	**1.14**	**-9.25**	0.10	-0.43	0.12	-2.40	0.70	-6.35	0.12	-0.51	0.89	-2.96	1.03	-8.12	0.08	-0.22	0.72
SlideShow	**-2.46**	**0.76**	**-9.32**	0.01	-0.01	0.10	-1.40	0.65	-8.87	0.02	-0.02	0.57	-1.78	0.71	-8.30	0.01	-0.01	0.09
ChinaSpeed	**-0.40**	**0.48**	**-6.57**	0.02	-0.09	0.21	-0.08	0.43	-6.12	0.05	-0.16	0.30	-0.11	0.40	-5.66	0.05	-0.02	0.31
BasketBallDrillText	**-0.41**	**0.47**	**-4.87**	0.03	-0.03	0.07	-0.22	0.38	-4.54	0.04	-0.13	0.28	-0.33	0.41	-4.77	0.04	-0.13	0.22
KristenAndSaraScreen	**-2.51**	**0.74**	**-7.85**	0.44	-0.31	0.61	-1.70	0.62	-6.91	0.51	-0.31	0.77	-2.09	0.43	-4.54	0.34	-0.35	0.69
BigBuckBunnyStudio	**-5.63**	**0.56**	**-3.86**	0.73	-0.17	0.18	-4.01	0.30	-3.24	0.84	-0.23	0.29	-4.68	0.46	-4.82	0.78	-0.22	0.27
ClearTypeSpreadsheet	**-4.79**	**1.12**	**-7.12**	0.18	-0.21	0.33	-0.36	1.00	-5.77	0.21	-0.28	0.45	-1.92	0.49	-3.03	0.23	-0.22	0.42
EnglishDocumentEditing	**-2.08**	**0.36**	**-11.49**	0.22	-0.37	0.36	-1.00	0.22	-1.17	0.28	-0.41	0.38	-2.21	0.26	-2.19	0.27	-0.43	0.32
CircuitLayoutPresentation	**-1.87**	**1.04**	**-6.65**	0.58	-0.25	0.63	-0.77	0.25	-0.88	0.62	-0.31	0.77	-1.14	0.68	-4.60	0.64	-0.33	0.79
ChineseDocumentEditing	**-1.82**	**0.88**	**-8.04**	0.67	-0.15	0.63	-0.94	0.46	-4.03	1.03	-0.21	0.72	-1.48	0.47	-4.13	1.01	-0.18	0.69
4:2:0-YUV Average	**-2.67**	**1.01**	**-10.53**	0.25	-0.16	0.34	-1.66	0.54	-5.98	0.36	-0.24	0.48	-1.97	0.65	-6.63	0.33	-0.21	0.43
Overall Average	**-2.93**	**1.02**	**-9.07**	0.27	-0.17	0.30	-2.07	0.53	-5.00	0.39	-0.27	0.42	-2.41	0.71	-6.03	0.35	-0.23	0.37

*10-bit sequence.

TABLE V: BD-BRE, BD-PSNR and BD-BR values of the test sequences under the LD-B configuration.

Sequence	Proposed RC method						RC method in [35]						RC method in [37]					
	vs. R-λ (HEVC)			vs. Baseline			vs. R-λ (HEVC)			vs. Baseline			vs. R-λ (HEVC)			vs. Baseline		
	BD-BRE	BD-PSNR	BD-BR	BD-BRE	BD-PSNR	BD-BR	BD-BRE	BD-PSNR	BD-BR	BD-BRE	BD-PSNR	BD-BR	BD-BRE	BD-PSNR	BD-BR	BD-BRE	BD-PSNR	BD-BR
4:4:4-RGB sequences																		
VenueVu	**-0.81**	**0.86**	**-23.65**	0.13	-0.55	1.33	-0.69	0.68	-16.83	0.02	-0.83	1.79	-0.70	0.64	-12.68	0.03	-0.70	1.43
FlyingGraphics	**-0.01**	**1.18**	**-20.83**	0.02	-0.01	1.06	0.07	0.78	-7.46	0.56	-0.11	1.64	0.00	0.87	-10.43	0.05	-0.01	0.96
Desktop	**-1.86**	**0.94**	**-1.05**	0.67	-0.52	1.12	-0.92	0.88	-1.01	0.72	-0.83	1.43	-1.43	0.11	-0.01	0.71	-0.73	1.38
Console	**-0.78**	**1.25**	**-9.41**	0.39	-0.29	1.36	-0.77	0.89	-7.18	0.49	-0.56	1.86	-0.74	0.83	-8.97	0.40	-0.50	1.64
MissionControlClip3	**-5.11**	**1.26**	**-18.73**	0.95	-0.43	0.93	-4.98	0.95	-13.47	1.02	-0.83	1.21	-4.98	0.92	-14.13	1.09	-0.77	1.07
EBURainFruits*	**-1.99**	**0.18**	**-5.41**	0.72	-0.18	1.21	-1.81	0.16	-4.55	0.93	-0.25	1.45	-1.92	0.13	-3.70	0.84	-0.21	1.37
Kimono1*	**-0.73**	**0.16**	**-10.41**	0.65	-0.19	0.95	-0.60	0.15	-8.46	0.77	-0.21	1.01	-0.63	0.12	-6.72	0.72	-0.23	0.97
Map	**-0.45**	**1.14**	-11.95	0.75	-0.01	0.58	-0.35	0.17	-12.22	1.20	-0.14	1.09	-0.32	0.72	**-12.90**	1.17	-0.11	0.76
Robot	**-0.14**	**1.06**	**-10.33**	0.69	-0.03	0.47	-0.01	0.45	-4.12	1.52	-0.95	0.92	-0.11	0.61	-7.48	1.18	-0.93	0.78
Viking	**-0.40**	**1.37**	-13.47	0.83	-0.30	0.75	-0.16	0.52	-12.22	0.77	-0.37	0.95	-0.20	0.86	**-18.21**	1.33	-0.32	0.84
WebBrowsing	**-9.42**	**1.02**	**-3.78**	0.12	-0.17	0.81	-8.21	0.30	-0.22	0.96	-0.79	1.47	-8.97	0.32	-1.41	0.11	-0.25	0.77
Programming	**-2.45**	**2.44**	**-21.52**	0.70	-0.42	0.45	-1.30	0.80	-13.44	0.99	-0.03	0.66	-2.15	1.13	-7.17	0.70	-0.69	0.37
SlideShow	-10.29	**1.51**	**-7.31**	0.09	-0.10	0.15	-10.20	1.47	-8.67	0.19	-0.18	0.23	**-10.42**	1.34	-6.96	0.13	-0.12	0.17
KristenAndSaraScreen	**-4.31**	**1.85**	**-20.99**	1.03	-0.51	1.09	-4.05	1.26	-15.50	1.18	-0.97	1.34	-4.18	1.27	-12.08	0.98	-0.58	1.13
BigBuckBunnyStudio	**-4.05**	**1.08**	**-2.36**	0.59	-0.34	0.76	-3.83	0.44	-0.12	0.73	-0.48	0.83	-3.14	1.01	-1.21	0.62	-0.41	0.88
ClearTypeSpreadsheet	**-10.98**	**0.88**	**-3.64**	0.18	-0.29	0.63	-10.62	0.55	-2.25	0.23	-0.34	0.69	-10.74	0.61	-2.02	0.19	-0.31	0.71
EnglishDocumentEditing	**-4.35**	**1.38**	**-6.38**	0.87	-0.34	1.04	-4.03	1.15	-4.77	1.03	-0.87	1.89	-4.08	1.10	-5.56	1.22	-0.64	1.68
CircuitLayoutPresentation	**-2.32**	**1.45**	**-13.13**	0.53	-0.63	0.72	-2.08	0.68	-7.09	0.84	-0.79	0.89	-2.14	0.89	-8.38	0.79	-0.74	0.84
ChineseDocumentEditing	**-1.59**	**1.04**	**-5.50**	1.06	-0.55	1.11	-1.31	0.73	-4.60	1.35	-0.96	1.14	-1.36	0.77	-4.10	1.38	-0.84	1.03
4:4:4-RGB Average	**-3.17**	**1.11**	**-10.33**	0.58	-0.33	0.86	-2.90	0.70	-6.77	0.78	-0.63	1.16	-2.96	0.76	-7.18	0.70	-0.50	1.04
4:4:4-YUV sequences																		
FlyingGraphics	**-0.03**	**1.22**	**-14.59**	0.03	-0.03	0.95	0.01	0.55	-4.62	0.61	-0.13	1.05	-0.02	0.57	-6.14	0.12	-0.04	1.01
Desktop	**-2.81**	**1.55**	**-2.73**	0.61	-0.49	1.04	-2.36	0.25	0.34	0.69	-0.75	1.37	-2.34	1.29	-1.90	0.59	-0.68	1.28
Console	**-1.31**	**1.32**	**-4.85**	0.42	-0.31	1.41	-0.96	0.98	-3.53	0.62	-0.73	1.97	-0.94	1.10	-3.85	0.61	-0.69	1.85
MissionControlClip3	**-4.61**	**2.42**	**-28.94**	0.82	-0.32	1.03	-4.12	1.10	-15.59	0.98	-0.79	1.27	-4.10	1.50	-22.85	1.02	-0.64	1.13
EBURainFruits*	**-2.16**	**0.21**	**-7.40**	0.67	-0.13	1.09	-1.98	0.17	-5.72	1.02	-0.53	1.25	-1.99	0.14	-4.94	0.72	-0.61	1.12
Kimono1*	**-0.45**	**0.59**	**-13.78**	0.71	-0.18	1.12	-0.31	0.16	-10.23	0.83	-0.23	1.19	-0.33	0.12	-8.37	0.79	-0.21	1.17
Map	**-0.63**	**1.08**	**-11.10**	0.81	-0.13	0.73	-0.58	0.16	-2.31	0.97	-0.67	1.07	-0.55	0.67	-6.60	1.03	-0.51	0.94
Robot	**-0.31**	**0.89**	**-7.68**	0.57	-0.11	0.53	-0.03	0.34	-2.45	0.78	-0.84	1.04	-0.06	0.56	-6.83	0.69	-0.73	0.82
WebBrowsing	**-6.67**	**1.20**	**-6.31**	0.15	-0.22	0.76	-6.28	0.99	-4.78	1.12	-0.83	1.17	-6.40	0.86	-3.85	0.17	-0.21	0.81
Programming	**-2.43**	**1.46**	**-19.63**	0.67	-0.53	0.63	-1.90	0.70	-11.44	0.87	-0.77	0.81	-2.25	0.90	-13.34	0.69	-0.65	0.74
SlideShow	-12.48	**1.36**	-2.62	0.11	-0.06	0.03	-12.27	0.88	-2.64	0.16	-0.12	0.12	**-12.64**	1.35	**-3.31**	0.10	-0.08	0.09
KristenAndSaraScreen	-3.64	**0.94**	**-15.93**	0.97	-0.47	1.04	-3.34	0.30	-3.39	1.13	-0.82	1.21	**-3.65**	0.27	-2.63	0.83	-0.49	1.09
BigBuckBunnyStudio	**-4.06**	1.11	**-2.83**	0.51	-0.54	0.92	-3.90	1.08	-1.06	0.62	-0.72	1.03	-3.94	**1.27**	-1.16	0.50	-0.51	1.12
ClearTypeSpreadsheet	**-7.11**	**0.93**	**-4.94**	0.24	-0.16	0.71	-6.52	0.52	-2.14	0.31	-0.23	0.73	-6.84	0.91	-3.25	0.33	-0.18	0.67
EnglishDocumentEditing	**-3.79**	**1.53**	-7.02	0.72	-0.28	0.96	-3.63	1.48	**-7.86**	0.89	-0.33	1.07	-3.63	1.31	-6.90	0.91	-0.38	1.12
CircuitLayoutPresentation	**-1.22**	**1.49**	**-17.09**	0.66	-0.89	0.92	-0.91	0.86	-9.64	0.83	-0.91	1.13	-1.12	1.13	-12.10	0.72	-0.89	1.08
ChineseDocumentEditing	**-3.53**	**1.03**	**-7.90**	1.32	-0.73	1.25	-2.83	0.77	-6.78	1.77	-1.05	1.65	-3.43	0.93	-6.45	1.45	-0.97	1.43
4:4:4-YUV Average	**-3.48**	**1.25**	**-11.12**	0.59	-0.30	0.87	-3.10	0.64	-6.43	0.88	-0.53	1.16	-3.31	0.86	-7.19	0.68	-0.48	0.98
4:2:0-YUV sequences																		
MissionControlClip2	**-3.39**	**1.32**	**-22.41**	0.71	-0.15	2.20	-1.39	0.71	-17.05	0.97	-0.58	3.94	-1.44	0.73	-14.04	0.89	-0.38	2.80
BasketBallScreen	**-6.03**	**1.88**	**-29.60**	0.40	-0.29	1.25	-3.24	0.65	-9.68	0.63	-0.54	2.36	-4.78	0.90	-8.18	0.60	-0.48	1.26
FlyingGraphics	**-0.03**	**0.27**	**-4.48**	0.01	0.00	0.83	-0.01	0.23	-3.70	0.32	-0.09	0.69	-0.01	0.20	-3.49	0.07	-0.05	0.92
Desktop	**-2.02**	**1.56**	**-22.64**	0.55	-0.41	1.44	-1.14	0.06	-4.51	0.50	-0.70	1.42	-1.47	1.25	-15.51	0.47	-0.71	1.39
Console	**-0.84**	**1.14**	**-6.25**	0.27	-0.39	0.97	-0.73	0.69	-3.86	0.33	-0.53	1.15	-0.79	0.70	-4.22	0.31	-0.54	1.13
MissionControlClip3	**-2.66**	**1.70**	**-34.05**	0.78	-0.27	1.20	-1.80	0.74	-15.55	0.82	-0.67	1.13	-1.97	1.01	-1.36	1.21	-0.51	1.09
Map	**-0.95**	**1.08**	**-16.30**	0.86	-0.30	0.65	-0.85	0.14	-3.12	0.92	-0.43	0.96	-0.81	0.72	-10.64	0.90	-0.53	0.93
Robot	**-0.40**	**1.07**	**-11.22**	0.71	-0.16	0.51	-0.24	0.46	-5.31	1.47	-0.91	1.03	-0.25	0.61	-4.49	1.13	-0.83	0.98
WebBrowsing	**-3.33**	**0.77**	**-3.54**	0.19	-0.28	1.11	-2.94	0.54	-3.26	1.07	-0.85	1.28	-3.06	0.40	-1.75	0.20	-0.48	1.13
Programming	**-1.13**	**1.47**	**-25.30**	0.81	-0.53	0.33	-1.06	0.83	-14.83	0.85	-0.61	0.41	-1.00	1.06	-17.99	0.79	-0.57	0.45
SlideShow	**-10.30**	1.07	**-20.42**	0.06	-0.01	-0.44	-8.23	0.26	-13.50	0.23	-0.01	0.72	-9.74	**1.10**	-19.88	0.26	-0.03	0.46
ChinaSpeed	**-0.03**	**1.58**	**-15.31**	0.02	-0.52	0.94	0.00	0.78	-5.34	0.08	-0.89	1.16	-0.02	1.34	-12.36	0.09	-1.18	1.07
BasketBallDrillText	**-0.05**	**1.27**	**-17.73**	0.04	-0.11	0.35	0.00	0.28	-13.90	0.07	-0.54	1.40	-0.03	0.78	-17.11	0.07	-0.64	0.44
KristenAndSaraScreen	**-3.66**	**0.24**	-2.45	1.12	-0.67	1.19	-3.36	0.20	**-3.09**	1.18	-0.73	1.27	-3.25	0.13	-2.21	1.09	-0.69	1.22
BigBuckBunnyStudio	**-3.97**	**1.06**	**-3.06**	0.77	-0.63	1.12	-3.87	0.79	-1.54	0.83	-0.68	1.19	-3.93	0.74	-1.85	0.75	-0.61	1.13
ClearTypeSpreadsheet	**-6.86**	**0.98**	**-4.53**	0.69	-0.43	0.93	-6.63	0.87	-3.24	0.73	-0.56	1.02	-6.81	0.96	-3.16	0.73	-0.49	1.13
EnglishDocumentEditing	**-1.42**	**1.27**	**-9.59**	0.92	-0.56	1.17	-1.16	0.88	-5.06	1.13	-0.79	1.43	-1.08	1.10	-5.74	1.05	-0.72	1.28
CircuitLayoutPresentation	**-0.74**	**1.10**	**-17.82**	0.57	-0.74	1.04	-0.71	0.78	-12.84	0.67	-0.87	1.29	-0.70	0.99	-15.52	0.63	-0.81	1.14
ChineseDocumentEditing	**-2.41**	**1.05**	**-12.92**	1.27	-0.67	1.18	-1.47	0.39	-5.40	1.59	-0.95	2.03	-2.06	0.49	-7.34	1.22	-0.72	1.67
4:2:0-YUV Average	**-2.64**	**1.15**	**-14.72**	0.57	-0.38	0.93	-2.04	0.54	-7.62	0.76	-0.63	1.36	-2.27	0.80	-8.78	0.66	-0.58	1.14
Overall Average	**-3.08**	**1.17**	**-12.09**	0.58	-0.34	0.89	-2.67	0.63	-6.96	0.80	-0.60	1.23	-2.83	0.81	-7.73	0.68	-0.52	1.05

* 10-bit sequence.

TABLE VI: BD-BRE, BD-PSNR and BD-BR values of the test sequences under the RA configuration.

Sequence	Proposed RC method						RC method in [35]						RC method in [37]					
	vs. R-λ (HEVC)			vs. Baseline			vs. R-λ (HEVC)			vs. Baseline			vs. R-λ (HEVC)			vs. Baseline		
	BD-BRE	BD-PSNR	BD-BR	BD-BRE	BD-PSNR	BD-BR	BD-BRE	BD-PSNR	BD-BR	BD-BRE	BD-PSNR	BD-BR	BD-BRE	BD-PSNR	BD-BR	BD-BRE	BD-PSNR	BD-BR
4:4:4-RGB sequences																		
VenueVu	**-0.13**	**0.76**	**-19.68**	0.15	-0.70	0.08	-0.01	0.74	-17.04	0.03	-0.86	0.06	-0.01	0.74	-15.56	0.02	-0.75	0.08
FlyingGraphics	**-12.37**	**1.37**	-13.26	0.04	-0.01	0.09	-5.75	0.88	-9.92	0.53	-0.14	0.09	-10.90	1.25	**-14.07**	0.06	-0.01	0.08
Desktop	**-3.28**	**0.35**	**-0.63**	0.77	-0.36	0.48	-3.03	0.29	-0.19	0.79	-0.43	0.62	-3.08	0.25	-0.17	0.72	-0.41	0.55
Console	**-1.66**	**1.05**	-8.99	0.59	-0.46	1.18	-1.12	0.61	-1.12	0.54	-0.66	1.42	-1.39	0.93	**-15.69**	0.49	-0.52	1.57
MissionControlClip3	**-3.30**	**1.48**	**-17.05**	1.15	-0.50	1.13	-2.88	1.16	-14.11	1.29	-0.61	1.18	-2.90	1.13	-11.11	1.25	-0.60	1.15
EBURainFruits*	**-1.34**	**0.32**	**-8.63**	0.09	-0.04	0.43	-1.17	0.22	-5.93	0.11	-0.05	0.48	-1.31	0.26	-7.10	0.12	-0.04	0.47
Kimono1*	**-0.29**	**0.14**	**-9.90**	0.03	0.01	0.11	-0.07	0.02	-1.10	0.06	0.00	0.16	-0.14	0.11	-4.73	0.04	0.00	0.15
Map	**-36.23**	**1.02**	**-24.42**	1.04	-0.08	0.13	-30.05	0.22	-12.39	1.25	-0.16	0.20	-30.26	0.51	-13.95	1.54	-0.13	0.14
Robot	**-0.26**	**0.88**	**-24.89**	0.68	-0.03	0.17	-0.02	0.26	-10.76	1.13	-0.82	0.22	**-0.26**	0.51	-3.07	0.97	-0.81	0.18
Viking	**-15.54**	**1.11**	-11.62	1.11	-0.37	0.05	**-12.60**	0.48	-12.58	0.98	-0.49	0.05	-13.88	1.02	-10.13	1.10	-0.41	0.05
WebBrowsing	**-1.99**	**1.11**	**-4.53**	0.09	-0.25	0.81	-0.95	0.14	-3.61	1.09	-0.95	1.53	-1.76	0.99	-3.77	0.10	-0.25	0.95
Programming	**-2.21**	**2.84**	**-14.73**	0.98	-0.66	0.09	-0.99	0.70	-11.79	0.97	-0.03	0.07	-1.70	1.60	-11.63	0.80	-0.80	0.07
SlideShow	**-6.02**	**0.96**	**-8.10**	0.26	-0.19	0.28	-5.68	0.10	-1.71	0.29	-0.33	0.36	-5.73	0.51	-6.75	0.22	-0.27	0.39
KristenAndSaraScreen	**-1.95**	**1.62**	**-17.47**	1.17	-0.64	0.97	-1.28	1.17	-13.66	1.34	-0.89	1.09	-1.74	1.57	-15.54	1.22	-0.73	1.01
BigBuckBunnyStudio	**-12.86**	**0.97**	**-5.31**	1.02	-0.47	0.18	-11.94	0.01	-0.31	1.18	-0.71	0.27	-12.31	0.85	-5.21	1.13	-0.59	0.21
ClearTypeSpreadsheet	**-3.12**	**1.18**	-1.97	0.59	-0.28	0.86	-2.71	0.31	-1.45	0.72	-0.37	0.94	-2.96	1.12	**-2.26**	0.69	-0.34	0.91
EnglishDocumentEditing	**-0.96**	**1.18**	**-3.84**	0.97	-0.69	0.14	-0.31	0.88	-3.14	1.12	-0.98	0.28	-0.40	0.82	-3.28	1.05	-0.74	0.18
CircuitLayoutPresentation	**-2.66**	**3.00**	**-20.82**	1.09	-0.66	0.37	-2.59	1.48	-10.28	1.27	-0.82	0.42	-2.47	2.10	-15.12	1.14	-0.74	0.43
ChineseDocumentEditing	**-3.61**	**1.70**	**-5.85**	0.98	-0.93	1.01	-3.08	0.70	-3.91	1.25	-0.99	1.23	-2.81	0.83	-4.08	1.13	-0.95	1.16
4:4:4-RGB Average	**-5.95**	**1.11**	**-10.34**	0.69	-0.40	0.48	-4.68	0.45	-5.52	0.84	-0.52	0.59	-5.21	0.83	-6.96	0.75	-0.47	0.55
4:4:4-YUV sequences																		
FlyingGraphics	**-14.42**	**1.15**	**-10.67**	0.06	0.00	0.21	-10.00	0.05	-2.02	0.78	-0.11	0.23	-12.60	0.78	-10.34	0.07	-0.04	0.25
Desktop	**-2.44**	**0.65**	**-1.53**	0.71	-0.49	0.63	-1.66	0.41	-1.03	0.85	-0.55	0.71	-2.20	0.28	-0.50	0.77	-0.51	0.64
Console	**-2.37**	**0.57**	**-2.30**	0.64	-0.53	1.22	-1.33	0.17	-0.60	0.82	-0.72	1.56	-1.99	0.48	-2.01	0.75	-0.69	1.49
MissionControlClip3	**-3.82**	**1.62**	**-18.65**	1.01	-0.46	1.11	-2.98	1.08	-13.15	1.28	-0.53	1.34	-3.72	1.13	-11.56	1.19	-0.51	1.23
EBURainFruits*	**-1.23**	**0.41**	**-11.92**	0.07	-0.01	0.37	-1.03	0.27	-7.41	0.09	-0.03	0.41	-1.07	0.29	-8.45	0.07	-0.02	0.40
Kimono1*	**-0.27**	**0.15**	**-11.36**	0.04	0.00	0.13	-0.17	0.03	-0.94	0.07	-0.02	0.15	-0.25	0.13	-8.28	0.05	-0.01	0.14
Map	**-34.62**	**0.11**	**-1.01**	1.23	-0.11	0.19	-26.75	0.00	-0.92	1.37	-0.18	0.25	-29.71	0.03	-0.51	1.40	-0.14	0.21
Robot	**-0.73**	**0.76**	**-18.07**	0.71	-0.09	0.22	-0.50	0.21	-6.93	0.92	-0.15	0.36	-0.45	0.49	-7.03	0.91	-0.13	0.31
WebBrowsing	**-2.93**	**1.11**	**-4.53**	0.12	-0.34	1.03	-0.95	0.14	-3.61	0.21	-0.53	1.24	-1.76	0.99	-3.77	0.17	-0.47	1.17
Programming	**-2.94**	**2.64**	**-28.94**	1.03	-0.72	0.21	-2.10	0.41	-5.33	1.35	-0.63	0.34	-2.77	1.67	-7.54	1.13	-0.82	0.28
SlideShow	**-7.12**	**1.10**	**-10.90**	0.23	-0.11	0.19	-6.64	0.12	-1.85	0.47	-0.13	0.27	-6.73	0.46	-6.36	0.39	-0.15	0.18
KristenAndSaraScreen	**-1.56**	**1.53**	**-14.51**	0.97	-0.59	1.01	-0.93	0.40	-2.38	1.07	-0.74	1.10	-1.46	0.11	-0.01	1.04	-0.63	1.05
BigBuckBunnyStudio	**-8.83**	**0.94**	**-4.80**	0.87	-0.41	0.19	-8.38	0.01	-0.24	1.16	-0.57	0.29	-8.70	0.71	-3.12	1.01	-0.55	0.20
ClearTypeSpreadsheet	**-3.81**	**1.04**	**-3.55**	0.61	-0.25	0.73	-3.20	0.59	-2.12	0.70	-0.31	0.79	-3.69	0.58	-1.17	0.71	-0.29	0.77
EnglishDocumentEditing	**-0.94**	**0.63**	**-6.39**	1.01	-0.53	0.18	-0.40	0.16	-0.87	1.19	-0.74	0.22	-0.47	0.31	-2.86	1.21	-0.61	0.19
CircuitLayoutPresentation	**-2.71**	**1.50**	-5.46	0.95	-0.57	0.41	-2.27	0.31	-2.09	1.07	-0.82	0.53	-2.46	**1.50**	**-6.59**	1.00	-0.69	0.49
ChineseDocumentEditing	**-3.20**	**1.72**	**-8.69**	1.07	-0.83	1.22	-2.81	1.27	-6.24	1.22	-0.98	1.35	-2.82	1.27	-5.59	1.21	-0.97	1.31
4:4:4-YUV Average	**-5.33**	**1.15**	**-11.09**	0.64	-0.34	0.51	-4.08	0.44	-5.17	0.86	-0.48	0.63	-4.70	0.74	-6.86	0.74	-0.44	0.56
4:2:0-YUV sequences																		
MissionControlClip2	**-2.11**	**1.27**	**-17.48**	0.81	-0.18	1.12	-1.15	0.58	-13.05	1.10	-0.48	1.02	-1.29	0.65	-13.59	0.88	-0.39	0.91
BasketBallScreen	**-2.29**	**1.56**	-23.52	0.46	-0.37	0.71	-0.74	0.98	-17.72	0.76	-0.64	0.75	-1.88	1.45	**-24.14**	0.66	-0.58	0.74
FlyingGraphics	**-8.33**	**1.28**	**-15.17**	0.10	-0.08	0.53	-6.84	0.20	-2.94	0.22	-0.13	0.64	-7.44	0.72	-10.64	0.15	-0.11	0.58
Desktop	**-1.10**	**0.81**	**-7.91**	0.68	-0.58	0.50	-0.65	0.47	-6.04	0.61	-0.89	0.42	-0.61	0.48	-4.46	0.47	-0.72	0.39
Console	**-1.82**	0.63	**-3.83**	0.42	-0.22	0.85	-1.47	0.48	-2.23	0.49	-0.35	0.93	-1.54	**0.68**	-3.78	0.51	-0.29	0.92
MissionControlClip3	**-2.62**	**1.50**	**-22.45**	0.95	-0.34	0.88	-2.25	0.99	-18.84	0.92	-0.66	0.68	-0.69	1.05	-14.39	0.97	-0.38	0.56
Map	**-31.19**	**0.44**	**-0.70**	0.94	-0.10	0.18	-23.74	0.02	-0.42	1.04	-0.18	0.26	-28.54	0.00	-0.30	1.10	-0.11	0.22
Robot	**-0.67**	**0.89**	**-24.62**	0.55	-0.05	0.11	-0.46	0.26	-10.13	0.62	-0.77	0.28	-0.53	0.49	-1.66	0.61	-0.58	0.27
WebBrowsing	**-1.89**	**1.10**	-3.20	0.10	-0.18	0.77	-1.29	0.43	-2.82	0.97	-0.82	0.94	-1.49	0.34	**-3.42**	0.13	-0.22	0.97
Programming	**-1.80**	**2.97**	**-30.64**	0.67	-0.43	0.12	-1.62	0.63	-9.65	0.89	-0.59	0.14	-1.60	0.83	-10.33	0.80	-0.52	0.14
SlideShow	**-5.86**	**1.05**	**-14.39**	-0.10	0.01	-0.15	-3.98	0.25	-11.91	0.27	-0.01	0.65	-5.34	0.69	-12.28	0.21	-0.03	0.14
ChinaSpeed	**-0.15**	**1.28**	**-12.33**	0.02	-0.46	0.34	-0.04	0.67	-4.63	0.06	-0.85	0.50	-0.04	0.72	-3.08	0.06	-0.07	0.51
BasketBallDrillText	**-0.25**	**1.35**	**-4.55**	0.06	-0.16	0.11	-0.01	0.04	-0.89	0.07	-0.69	0.47	-0.24	1.15	-4.11	0.08	-0.71	0.37
KristenAndSaraScreen	**-2.34**	**1.65**	**-12.42**	1.02	-0.43	0.34	-1.80	0.62	-4.28	1.18	-0.60	0.48	-1.99	1.27	-8.54	1.07	-0.59	0.44
BigBuckBunnyStudio	**-7.32**	**0.93**	**-5.20**	0.73	-0.36	0.20	-7.02	0.01	-0.14	0.87	-0.51	0.29	-7.22	0.80	-2.53	0.81	-0.47	0.27
ClearTypeSpreadsheet	**-1.69**	**1.09**	**-6.18**	0.44	-0.11	0.58	-1.03	0.65	-5.02	0.59	-0.24	0.60	-1.27	0.81	-5.16	0.53	-0.17	0.61
EnglishDocumentEditing	**-0.90**	**0.65**	**-14.52**	0.95	-0.49	0.20	-0.57	0.28	-8.34	1.03	-0.63	0.28	-0.63	0.45	-10.85	1.01	-0.57	0.26
CircuitLayoutPresentation	**-2.58**	**0.43**	**-6.46**	0.83	-0.41	0.38	-2.39	0.30	-1.95	0.98	-0.69	0.49	-2.37	0.22	-3.11	0.99	-0.62	0.44
ChineseDocumentEditing	**-1.71**	**1.52**	**-13.18**	1.32	-0.73	1.03	-1.33	0.76	-9.34	1.89	-0.93	1.03	-1.29	1.01	-8.10	1.74	-0.73	1.02
4:2:0-YUV Average	**-4.03**	**1.18**	**-12.57**	0.58	-0.30	0.46	-3.07	0.45	-6.86	0.77	-0.56	0.57	-3.47	0.73	-7.60	0.67	-0.41	0.51
Overall Average	**-5.19**	**1.15**	**-11.19**	0.65	-0.34	0.49	-4.01	0.44	-5.67	0.84	-0.52	0.60	-4.53	0.77	-7.00	0.73	-0.43	0.55

*10-bit sequence.

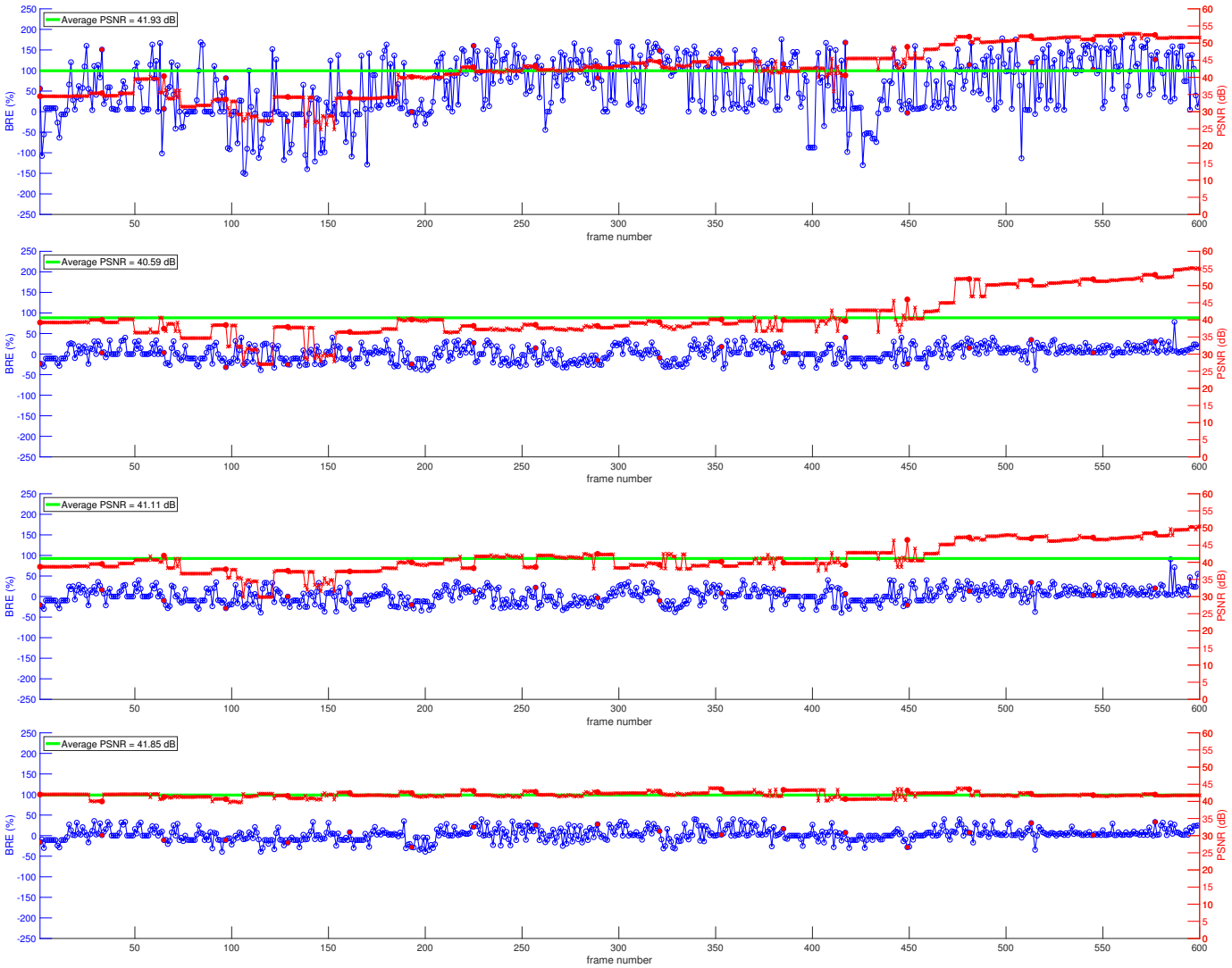

Fig. 4: Per-frame BRE and PSNR values for the 4:4:4-RGB sequence *Map* coded at $R_{target} = 0.82$ Mbps (QP=32) using (1^{st}row) the RC method of H.265/HEVC, (2^{nd}row) the method in [35], (3^{rd}row) the method in [37], and (4^{th}row) the proposed RC method. The sequences are coded using the RA configuration. Solid red dots represent intra-predicted frames, while clear dots represent inter-predicted frames.

appear blurred. Interestingly, the visual quality attained by the RC methods in [35] and [37], and the proposed RC method are similar for the *Map* frame. However, the proposed RC method attains the highest PSNR values. Namely, the PSNR values attained by the RC method of H.265/HEVC, the RC methods in [35] and [37], and the proposed RC method are, respectively, $\{35.34, 40.17, 39.66, 42.04\}$ dB. For the case of the *MissionControlClip2* frame, the visual quality attained by the proposed RC method is the highest despite the low target bit rate. Note that it is possible to clearly distinguish the text and the edges in the plot region of the frame reconstructed after compression by the proposed RC method. The PSNR values attained by the RC method of H.265/HEVC, the RC methods in [35] and [37], and the proposed RC method on this frame are, respectively, $\{29.77, 31.15, 31.55, 34.97\}$ dB. The highest reconstruction quality attained by the proposed RC method comes as a consequence of the low per-frame BER values and the consistent reconstruction quality achieved on those frames previously coded, as shown in Figs. 4 and 5.

A. *Performance on NC sequences*

It is important to mention that although the proposed RC method is designed for SC sequences, it can also be used for other type of content [25, 48, 49] including NC sequences. To evaluate its performance on NC sequences, all Class A and Class B sequences of the common test conditions [50] are first encoded at the four fixed QPs $\{37, 32, 27, 22\}$. The actual bit rates used to compress the video sequences at these four fixed QPs are then set as the target bit rates for RC. We evaluate our proposed RC method against the RC method of the H.265/HEVC reference implementation (HM16.9 [43]). Table VII tabulates per-class average BD-BRE, BD-PSNR, and BD-BR values. Our proposed RC method achieves improvements in RC accuracy, as well as PSNR gains and BR reductions. As expected, these improvements are not as important as in the case of SC sequences, as the R-λ model used for RC by the H.265/HEVC reference implementation has parameters trained on NC sequences. Hence, it is expected that this RC method

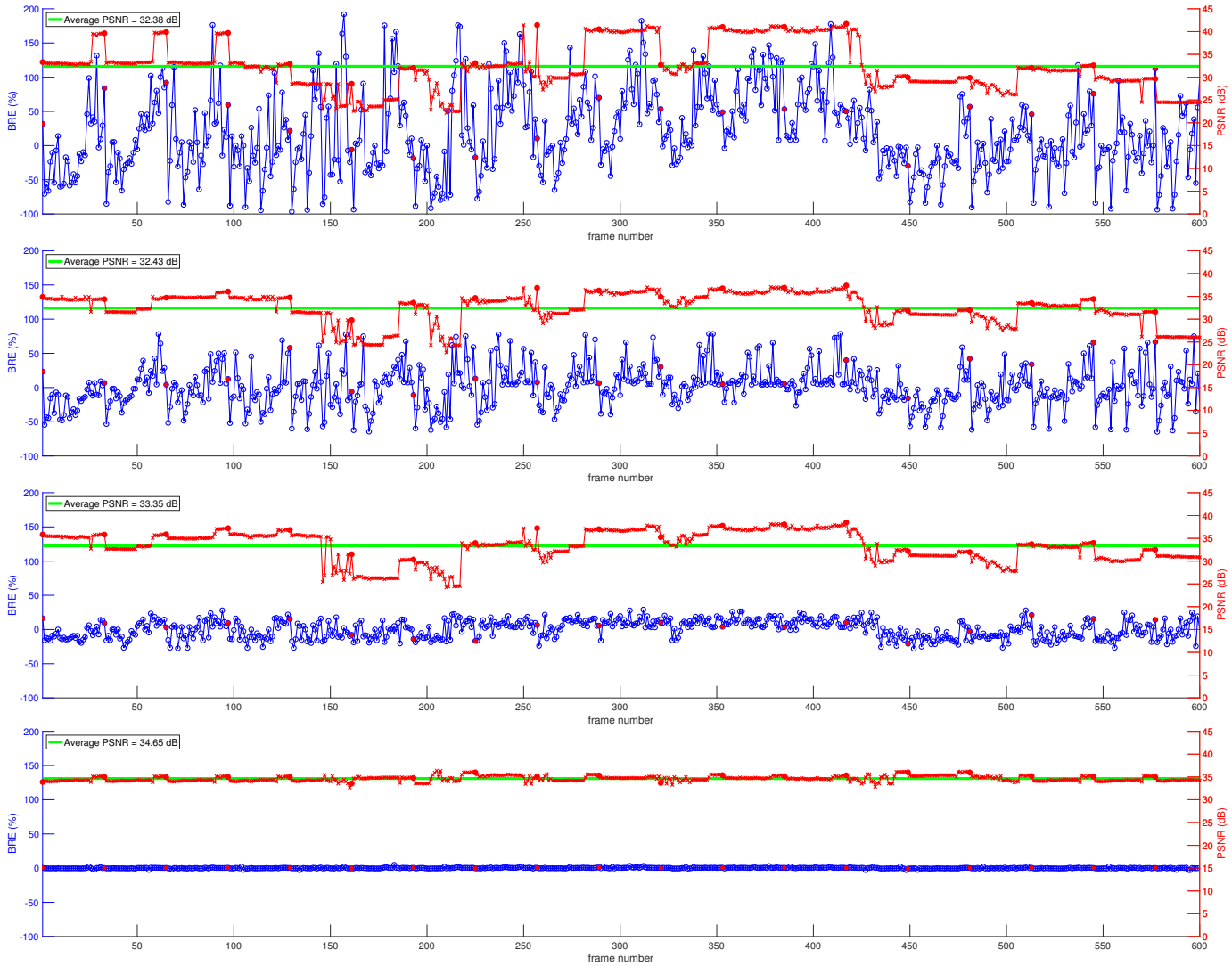

Fig. 5: Per-frame BRE and PSNR values for the 4:2:0-YUV sequence *MissionControlClip2* coded at $R_{target} = 1.11$ Mbps (QP=37) using (1^{st}row) the RC method of H.265/HEVC, (2^{nd}row) the method in [35], (3^{rd}row) the method in [37], and (4^{th}row) the proposed RC method. The sequences are coded using the RA configuration. Solid red dots represent intra-predicted frames, while clear dots represent inter-predicted frames.

performs strongly on these type of sequences.

TABLE VII: Average BD-BRE, BD-PNSR and BD-BRE values of the proposed RC method compared to the RC method in H.265/HEVC for Class A and Class B sequences with target bit rates based on fixed QP values.

Class	AI			LD-B			RA		
	BD-BRE	BD-PSNR	BD-BR	BD-BRE	BD-PSNR	BD-BR	BD-BRE	BD-PSNR	BD-BR
Class A	-0.45	0.75	-4.32	-0.23	0.39	-2.57	-0.34	0.51	-2.79
Class B	-0.39	0.62	-5.03	-0.31	0.27	-1.35	-0.41	0.46	-1.95
Average	-0.42	0.69	-4.68	-0.27	0.33	-1.96	-0.38	0.49	-2.37

B. Computational Complexity

Let us recall that the proposed method requires simple linear operations to approximate λ and QP values (see Eqs. 12 and 13), which means that its complexity is linear. Moreover, computing coding costs of blocks does not require further operations when our method is embedded in a standard H.265/HEVC encoder, as it relies on the Hadamard Transform for intra-predicted frames and on the residual signal for inter-predicted frames. The Hadamard Transform is computed during intra-coding for mode selection, while the residual signal is computed during inter-coding. Our proposed RC method is, however, expected to increase coding times mainly due to the search for previously coded blocks that fulfil the two criteria to be selected as control points. To minimize the increase in coding times due to this search, our method computes the coding cost of an LCU only once. Further, the set of possible control points are organized in an array in descending order according to their cost. Searching for the control points for the current LCU then reduces to finding elements in a sorted array, whose complexity is further reduced by keeping pointers to the location of the entries in such a sorted array with the maximum, minimum and mid costs.

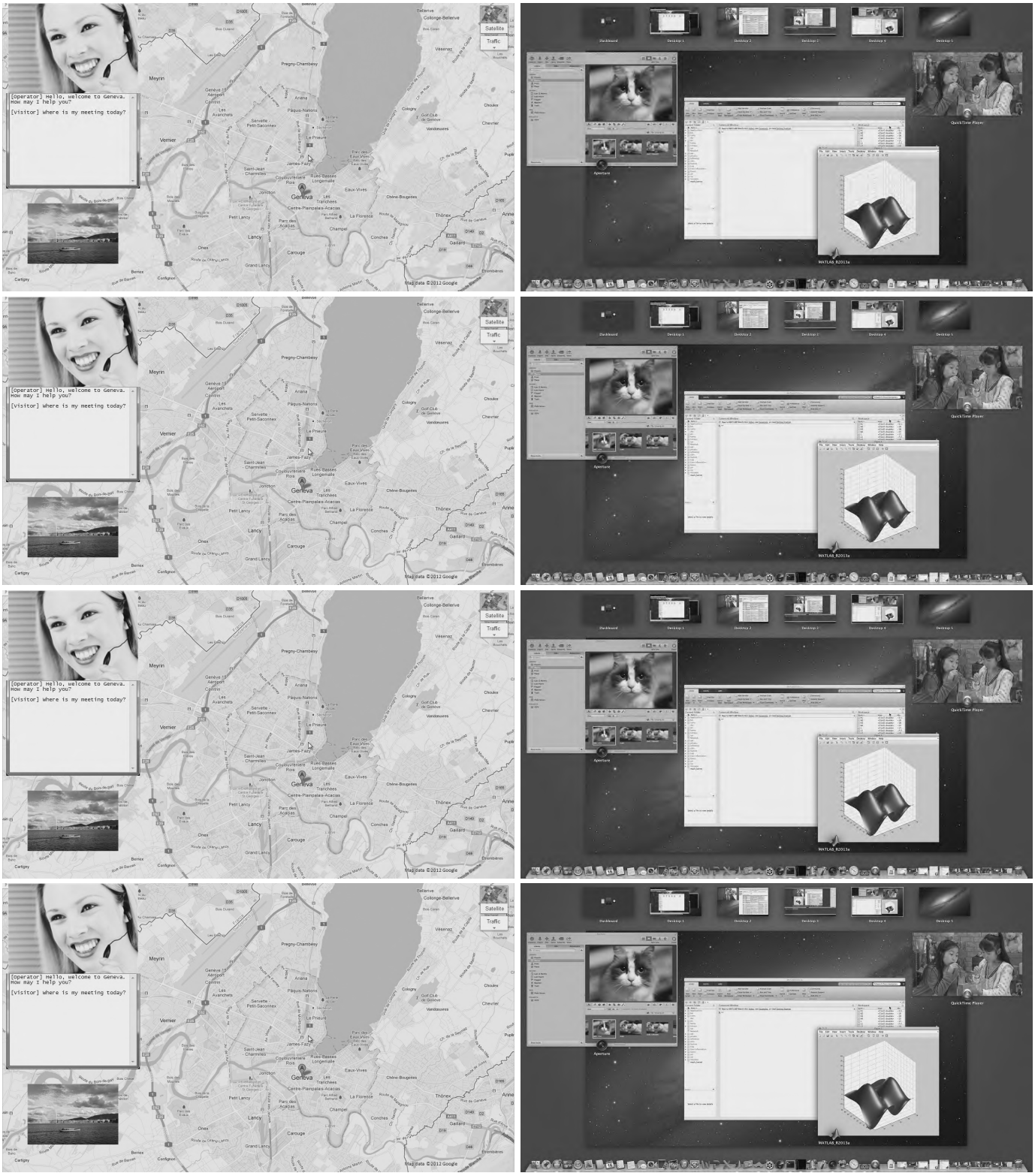

Fig. 6: Subjective quality of (left column) frame 46 of the 4:4:4-RGB sequence *Map* coded at $R_{target} = 0.82$ Mbps (QP= 32), and (right column) frame 263 of the 4:2:0-YUV sequence *MissionControlClip2* coded at $R_{target} = 1.11$ Mbps (QP= 37). Top to bottom rows depict results for the RC method of H.265/HEVC, the RC method in [35], the one in [37], and the proposed RC method, respectively. The sequences are coded using the RA configuration with per-frame BRE and PSNR values as depicted in Figs. 4 and 5.

To estimate the computational complexity of the proposed RC method, we first compute the average computational time of a search for control points in such a sorted array. We use all test SC sequences and an Intel Core i7 CPU at 3.1 GHz and 16 GB of RAM. A search takes, on average, 0.0013 ms for a single block. Overall, our RC method increases coding times

by an average of 1.61% for all tested SC sequences and coding configurations compared to the RC method of H.265/HEVC. The RC methods proposed in [35] and [37] increase coding times by an average of 1.03% and 0.92%, respectively, for all tested SC sequences and coding configurations. The increase incurred by the method in [35] is due to the fast motion estimation employed to compute the complexity of frames and LCUs. The increase incurred by the method in [37], on the other hand, is due to the classification of CTUs into one of three classes: T-CTUs, Screen Image-CTUs and Nature Image-CTUs.

One advantage of our RC method, despite the small increase in coding times, is that it can be readily used in a standard H.265/HEVC encoder to encode any type of content without having to find a proper set of model parameters, $\theta = \{\alpha, \beta, a, q\}$, through training. Conversely, other RC methods that rely on these parameters would require a re-training of θ to improve BD-BRE performance for those contents that are very dissimilar to the natural content.

V. Conclusion

This paper presented a new low-complexity RC method within the context of predictive transform coding of SC sequences. We first showed that the R-D and R-QP relationships of a video sequence may be approximated by piecewise linear segments and RANSAC. By leveraging the low-complexity of such piecewise linear approximations and the availability of R-D and R-QP information within the current frame, we then proposed to approximate the R-D and R-QP curves of each block by linearly interpolating among $N \geq 2$ control points, each representing actual measured rate, distortion and QP values of previously coded blocks within the same frame. Therefore, the proposed RC method does not rely on any trained R-D or R-QP models, which makes it suitable for any type of video content and flexible enough for optimization of both, a target bit rate or a target reconstruction quality. Performance evaluations on several SC sequences showed that our RC method attains a better performance, in terms of PSNR and bit rate, than the RC method of H.265/HEVC and other methods specifically designed for SC sequences. Our results also showed that our RC method can achieve a more constant reconstruction quality on a per-frame basis with a small increase in coding times.

References

[1] B. Li, H. Li, L. Li, and J. Zhang, "λ-domain rate control algorithm for high efficiency video coding," *IEEE Transactions on Image Processing*, vol. 23, no. 9, pp. 3841-3854, 2014.

[2] F. Auli-Llinas, J. Serra-Sagrista, J. L. Monteagudo-Pereira, and J. Bartrina-R, "Efficient rate control for JPEG2000 coder and decoder," in *Data Compression Conference*, 2006, pp. 282-291.

[3] V. Sanchez, "Rate control for HEVC intra-coding based on piecewise linear approximations," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2018, pp. 1782-1786.

[4] A. Fiengo, G. Chierchia, M. Cagnazzo, and B. Pesquet-Popescu, "Rate allocation in predictive video coding using a convex optimization framework," *IEEE Transactions on Image Processing*, vol. 26, no. 1, pp. 479-489, 2016.

[5] H. Guo, C. Zhu, S. Li, and Y. Gao, "Optimal bit allocation at frame level for rate control in HEVC," *IEEE Transactions on Broadcasting*, vol. 65, no. 2, pp. 270-281, 2018.

[6] S. Li, M. Xu, Z. Wang, and X. Sun, "Optimal bit allocation for CTU level rate control in HEVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 11, pp. 2409-2424, 2016.

[7] Z. Chen and X. Pan, "An optimized rate control for low-delay H.265/HEVC," *IEEE Transactions on Image Processing*, vol. 28, no. 9, pp. 4541-4552, 2019.

[8] K. Ramchandran, A. Ortega, and M. Vetterli, "Bit allocation for dependent quantization with applications to multiresolution and MPEG video coders," *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 533-545, 1994.

[9] Y. Yang and S. S. Hemami, "Generalized rate-distortion optimization for motion-compensated video coders," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, no. 6, pp. 942-955, 2000.

[10] X. Liang, Q. Wang, Y. Zhou, B. Luo, and A. Men, "A novel RQ model based rate control scheme in HEVC," in *Visual Communications and Image Processing*, 2013, pp. 1-6.

[11] Z. He and S. K. Mitra, "Optimum bit allocation and accurate rate control for video coding via/spl ρ-domain source modeling," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 10, pp. 840-849, 2002.

[12] T. Chiang and Y.-Q. Zhang, "A new rate control scheme using quadratic rate distortion model," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 1, pp. 246-250, 1997.

[13] H. Choi, J. Yoo, J. Nam, D. Sim, and I. V. Bajić, "Pixel-wise unified rate-quantization model for multi-level rate control," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 6, pp. 1112-1123, 2013.

[14] H. Choi, J. Nam, J. Yoo, D. Sim, and I. Bajić, "Rate control based on unified RQ model for HEVC," ITU-T SG16 Contribution, JCTVC-H0213, pp. 1-13, 2012.

[15] M. Naccari and F. Pereira, "Quadratic modeling rate control in the emerging HEVC standard," in *IEEE Picture Coding Symposium*, 2012, pp. 401-404.

[16] J. Wei, B. Soong, and Z. Li, "A new rate-distortion model for video transmission using multiple logarithmic functions," *IEEE Signal Processing Letters*, vol. 11, no. 8, pp. 694-697, 2004.

[17] M. Santamaria, E. Izquierdo, S. Blasi, and M. Mrak, "Estimation of rate control parameters for video coding using CNN," in 2018 *IEEE Visual Communications and Image Processing*, 2018, pp. 1-4.

[18] Y. Li, B. Li, D. Liu, and Z. Chen, "A convolutional neural network-based approach to rate control in HEVC intra coding," in *IEEE Visual Communications and Image Processing*, 2017, pp. 1-4.

[19] G. J. Sullivan, J. R. Ohm, F. Bossen, and T. Wiegand, "JCTVC AHG report: HM subjective quality investigation," JCTVCH0022, 2012.

[20] K. R. Perez-Daniel and V. Sanchez, "Luma-aware multi-model rate-control for HDR content in HEVC," in *IEEE International Conference on Image Processing*, 2017, pp. 1022-1026.

[21] B. Li, H. Li, and L. Li, "JCT-VC M0036: Adaptive bit allocation for R-lambda model rate control in HM," *13th Meeting of Joint Collaborative Team on Video Coding of ITU-T SG 16 WP3*, 2013.

[22] M. R. Ardestani, A. A. B. Shirazi, and M. R. Hashemi, "Rate-distortion modeling for scalable video coding," in *17th International Conference on Telecommunications*, 2010, pp. 923-928.

[23] B. Li, D. Zhang, H. Li, and J. Xu, "QP determination by lambda value," in *JCTVC-I0426, 9th JCTVC Meeting*, 2012.

[24] J. Lainema, F. Bossen, W. Han, J. Min, and K. Ugur, "Intra coding of the HEVC standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1792-1801, 2012.

[25] V. Sanchez and M. Hernandez-Cabronero, "Graph-based rate control in pathology imaging with lossless region of interest coding," *IEEE Transactions on Medical Imaging*, vol. 37, no. 10, pp. 2211-2223, 2018.

[26] J. Xu, R. Joshi, and R. A. Cohen, "Overview of the emerging HEVC screen content coding extension," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 1, pp. 50-62, 2015.

[27] V. Sanchez, F. Auli-Llinas, and J. Serra-Sagrista, "DPCM-based edge prediction for lossless screen content coding in HEVC," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 6, no. 4, pp. 497-507, 2016.

[28] Q. Zhou, L. Zhao, K. Zhou, T. Lin, H. Wang, S. Wang, and M. Jiao, "String prediction for 4:2:0 format screen content coding and its implementation in AVS3," *IEEE Transactions on Multimedia*, pp. 1-1, 2020.

[29] Y. Yang, K. Zhou, L. Zhao, and T. Lin, "An ultralow complexity string matching approach to screen content coding in AVS3," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1-1, 2020.

[30] L. Zhao, K. Zhou, J. Guo, S. Wang, and T. Lin, "A universal string matching approach to screen content coding," *IEEE Transactions on Multimedia*, vol. 20, no. 4, pp. 796-809, 2018.

[31] L. Zhao, T. Lin, K. Zhou, S. Wang, and X. Chen, "Pseudo 2d string matching technique for high efficiency screen content coding," *IEEE Transactions on Multimedia*, vol. 18, no. 3, pp. 339-350, 2016.

[32] T. Lin, P. Zhang, S. Wang, K. Zhou, and X. Chen, "Mixed chroma sampling-rate high efficiency video coding for full-chroma screen content," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 1, pp. 173-185, 2013.

[33] M. Mrak and J. Xu, "Improving screen content coding in HEVC by transform skipping," in *2012 Proceedings of the 20th European Signal Processing Conference*, 2012, pp. 1209-1213.

[34] Y. Guo, B. Li, S. Sun, and J. Xu, "Rate control for screen content coding in HEVC," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2015, pp. 1118-1121.

[35] J. Xiao, B. Li, S. Sun, and J. Xu, "Rate control with delay constraint for screen content coding," in *2017 IEEE Visual Communications and Image Processing (VCIP)*, 2017, pp. 1- 4.

[36] S. Wang, J. Li, S. Wang, S. Ma, and W. Gao, "A frame level rate control algorithm for screen content coding," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2018, pp. 1-4.

[37] Y. Yang, L. Shen, H. Yang, and P. An, "A content-based rate control algorithm for screen content video coding," *Journal of Visual Communication and Image Representation*, vol. 60, pp. 328 - 338, 2019.

[38] G. J. Sullivan, J. Ohm, W. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649-1668, 2012.

[39] Y. Kim, D. Jun, S. Jung, and J. Choi, "A fast intra prediction method using Hadamard transform in high efficiency video coding," in *Visual Information Processing and Communication III*, vol. 8305, 2012, p. 83050A.

[40] F. Auli-Llinas, A. Bilgin, and M. W. Marcellin, "Fast rate allocation through steepest descent for JPEG2000 video transmission," *IEEE Transactions on Image Processing*, vol. 20, no. 4, pp. 1166-1173, 2010.

[41] L.-J. Lin and A. Ortega, "Bit-rate control using piecewise approximated rate-distortion characteristics," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no. 4, pp. 446-459, 1998.

[42] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381-395, 1981.

[43] High Efficiency Video Coding (HEVC) Software, HM16.9, Accessed June 16, 2020, https://hevc.hhi.fraunhofer.de/svn/svn HEVCSoftware/tags/HM-16.9/.

[44] H. Yu, R. Cohen, K. Rapaka, and J. Xu, "Common test conditions for screen content coding," *Joint Collaborative Team on Video Coding (JCTVC)*, vol. JCTVCT1015, 2015.

[45] V. Baroncini, H. Yu, R. Joshi, S. Liu, X. Xiu, and J. Xu, "Draft of final report on SCC verification test," *Joint Collaborative Team on Video Coding (JCTVC)*, vol. JCTVCAA0040, 2017.

[46] V. Sanchez, F. Auli-Llinas, R. Vanam, and J. Bartrina-Rapesta, "Rate control for lossless region of interest coding in HEVC intra-coding with applications to digital pathology images," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2015, pp. 1250-1254.

[47] J. Ribas-Corbera, P. A. Chou, and S. L. Regunathan, "A generalized hypothetical reference decoder for H.264/AVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 674-687, 2003.

[48] A. Madabhushi and G. Lee, "Image analysis and machine learning in digital pathology: Challenges and opportunities," 2016.

[49] V. Sanchez, P. Nasiopoulos, and R. Abugharbieh, "Efficient 4D motion compensated lossless compression of dynamic volumetric medical image data," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2008, pp. 549-552.

[50] F. Bossen et al., "Common test conditions and software reference configurations," *JCTVC-L1100*, vol. 12, 2013.